Investigação de algoritmos sequenciais de agrupamento com pré-processamento de dados em aprendizado de máquina

Eduardo Machado Real

Janeiro / 2014

Dissertação de Mestrado em Ciência da Computação

Investigação de algoritmos sequenciais de agrupamento com pré-processamento de dados em aprendizado de máquina

Esse documento corresponde à dissertação de mestrado apresentada à Banca Examinadora da Dissertação no curso de Mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista.

Campo Limpo Paulista, 13 de Janeiro de 2014.

Eduardo Machado Real

Profa. Dra. Maria do Carmo Nicoletti
Orientadora

Faculdade Campo Limpo Paulista
Programa de Mestrado em Ciência da Computação

"Investigação de Algoritmos Sequenciais de Agrupamento com Pré-Processamento de Dados em Aprendizado de Máquina"

EDUARDO MACHADO REAL

Dissertação de Mestrado apresentada ao Programa de Mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Membros da Banca:

Profa. Dra. Maria do Carmo Nicoletti
(Orientadora - FACCAMP)

Prof. Dr. José Hiroki Saito
(FACCAMP)

Profa. Dra. Ana Maria Monteiro
(FACCAMP)

Profa. Dra. Heloisa de Arruda Camargo
(DC/UFSCar)

Campo Limpo Paulista -SP

Janeiro/2014

FICHA CATALOGRÁFICA

Dados Internacionais de Catalogação na Publicação (CIP)

Câmara Brasileira do Livro, São Paulo, Brasil

Real, Eduardo Machado
Investigação de algoritmos sequenciais de agrupamento com pré-processamento de dados em aprendizado de máquina / Eduardo Machado Real. Campo Limpo Paulista, SP: FACCAMP, 2013.

Orientadora: Maria do Carmo Nicoletti.
Dissertação (mestrado) – Faculdade Campo Limpo Paulista – FACCAMP.

1. Algoritmos. 2. Programação de computadores. I. Nicoletti, Maria do Carmo.
II. Faculdade Campo Limpo Paulista. III. Título.

CDD-005.1

Agradecimentos

À Deus por ter me dado condições de lutar e alcançar os objetivos pretendidos.

À professora Dr. Maria do Carmo Nicoletti, pela orientação e, principalmente, pela dedicação, ensinamentos e incentivo durante o desenvolvimento desta dissertação.

Ao professor e coordenador de curso Dr. Osvaldo Luiz de Oliveira, pelo incentivo e pelo pronto atendimento e esclarecimentos a quaisquer dúvidas durante o curso.

Aos professores, funcionários e mestrandos do programa de mestrado em Ciência da Computação da Faculdade Campo Limpo Paulista.

Aos professores membros das bancas examinadoras pela disposição e contribuições ao trabalho examinado.

Aos professores, funcionários e à Universidade Estadual de Mato Grosso do Sul pela oportunidade, incentivo e confiança no meu trabalho desde o ano de 2009.

Àqueles que de um modo ou de outro contribuíram para a realização deste trabalho.

Resumo. *Esta dissertação tem como foco principal a investigação de algoritmos de aprendizado de máquina não supervisionados identificados como algoritmos sequenciais de agrupamento, particularmente o Basic Sequential Algorithmic Squeme (BSAS), o Modified Basic Sequential Algorithmic Squeme (MBSAS) e o Two-Threshold Sequential Algorithmic Squeme (TTSAS). Esses algoritmos produzem um único agrupamento e são, geralmente, bastante rápidos. Têm, entretanto, a desvantagem do resultado final ser, usualmente, dependente da ordem na qual os dados (a serem agrupados) são apresentados aos algoritmos e dos valores de parâmetros definidos pelo usuário. Além da investigação de tais algoritmos com vistas a minimizar impactos de sua desvantagem intrínseca, foram investigados e implementados estratégias de refinamento pós-agrupamento, técnicas de pré-processamento de dados e métodos de validação, como uma maneira de promover, refinar e validar o resultado do aprendizado. Todas as implementações desenvolvidas estão disponibilizadas no sistema computacional SEQ_CLUSTER, que oferece uma plataforma para uso, avaliação e testes tanto dos algoritmos sequenciais de agrupamento quanto dos dois procedimentos de refinamento, i.e., o merge e o reassignment. O trabalho apresenta e discute os resultados de experimentos realizados em conjuntos de dados do UCI Repository e em quatro conjuntos de dados sintéticos. Os resultados dos experimentos indicam que os algoritmos sequenciais geram bons resultados para a maioria dos conjuntos de dados investigado; no entanto a ordem em que os dados são processados e os valores dos parâmetros fornecidos pelo usuário podem ter uma forte influência nos resultados de agrupamento obtidos. Também foi detectado empiricamente que os resultados puderam ser melhorados pelas estratégias de refinamento, bem como pelo pré-processamento de dados.*

Abstract*: This dissertation is mainly focused on the investigation of unsupervised learning algorithms identified as sequential clustering algorithms, particularly the Basic Sequential Algorithr Squeme (BSAS), the Modified Basic Sequential Algorithmic Squeme (MBSAS) and the Two-Thresho Sequential Algorithmic Squeme (TTSAS). The three algorithms produce a single clustering and are, generally, quite fast. They have, however, the disadvantage of their final result be, usually, dependent on the order in which the data (to be clustered) are presented to the algorithms and, also, the parameter values defined by the user. In addition to investigating such algorithms in order to minimize the impacts of their intrinsic disadvantages, a few other techniques, such as post-clustering refinement strategies, data pre-processing and validation methods, were also implemented as a way to promote, refine and validate the result of learning. A computational system, named SEQ_CLUSTER, was developed for supporting the research work and the conducted experiments, based on datasets from the UCI repository as well as synthetic datasets. Overall, the experiments have shown that the algorithms induced good clustering results for most of data sets; however the order in which the data patterns are processed and the parameter values supplied by the user can have a strong influence on the clustering results obtained. It was also empirically detected that results can be further improved by the two refinement strategies as well as by pre-processing the data.*

Sumário

Lista de Tabelas

Lista de Figuras

Lista de Algoritmos

Introdução

Essa dissertação descreve a pesquisa em nível de mestrado intitulada "*Investigação de algoritmos sequenciais de agrupamento com pré-processamento de dados em aprendizado de máquina*", realizada por Eduardo Machado Real junto ao PMCC-FACCAMP, C. L. Paulista - SP, sob orientação da Profa. Maria do Carmo Nicoletti. A área de pesquisa de Aprendizado de Máquina (AM) é uma subárea da Inteligência artificial (IA) que, entre outros, investiga o desenvolvimento de formalismos e técnicas que permitem a construção de sistemas automáticos de aprendizado. Dentre as várias taxonomias propostas para organizar as muitas técnicas que implementam AM, aquela que adota como critério o nível da supervisão associado ao conjunto de treinamento, durante a fase de aprendizado, foi a de particular interesse para o desenvolvimento deste trabalho, especificamente o grupo de técnicas de aprendizado não-supervisisonado.

O trabalho de pesquisa realizado investigou três algoritmos de agrupamento caracterizados como sequenciais, com a agregação de técnicas de pré-processamento de dados (como uma maneira de tratar os dados a serem utilizados pelos métodos), estratégias de refinamento pós-agrupamento (com vistas a refinar os resultados obtidos) e, também, métodos de validação (para avaliar os resultados obtidos). A dissertação está organizada como segue:

Capítulo 1: contextualiza a área Aprendizado de Máquina (AM) na qual a pesquisa descrita neste documento se insere, apresentando suas principais características, objetivos, conceitos, modelos, etc.

Capítulo 2: apresenta um panorama atual da subárea de AM conhecida como *aprendizado não supervisionado* e dos algoritmos chamados de algoritmos de agrupamento que implementam tal tipo de aprendizado automático. Descreve uma taxonomia de métodos não supervisionados de aprendizado de máquina e, também, busca formalizar a notação a ser empregada nos capítulos seguintes.

Capítulo 3: apresenta e discute o algoritmo de agrupamento K-Means os chamados algoritmos de agrupamento sequenciais, a saber, K-Means, *Basic Sequential Algorithmic Scheme* (BSAS), *Modified Basic Sequential Algorithmic Scheme* (MBSAS) e *Two-Threshold Sequential Algorithmic Scheme* (TTSAS), bem como aborda estratégias de refinamento pós-agrupamento e brevemente alguns trabalhos na área que foram investigados no curso desta pesquisa.

Capítulo 4: discute dois aspectos relevantes relacionados a AM: (1) a importância do pré-processamento de dados, como um processo que antecede o uso de técnicas de AM, com vistas a tratar os dados disponibilizados ao aprendizado e (2) o processo de validação, no contexto de técnicas de agrupamento, por meio da apresentação e caracterização de três índices de validação que são comumente empregados em experimentos com algoritmos de agrupamento.

Capítulo 5: apresenta em linhas gerais a arquitetura e as diferentes funcionalidades disponibilizadas pelo sistema computacional chamado SEQ_CLUSTER. Tal sistema foi desenvolvido com o objetivo de disponibilizar uma plataforma para uso e experimentação com algoritmos sequenciais de agrupamento (incluindo, também, implementação de outro algoritmo de agrupamento, com vistas a possíveis comparações), bem como de ferramentas computacionais de pré-processamento de dados e de validação de resultados. O capítulo descreve a arquitetura geral do SEQ_CLUSTER, as integrações entre os seus subsistemas e as várias funcionalidades disponibilizadas pelo sistema.

Capítulo 6: descreve em detalhes um conjunto de experimentos relativos à tarefa de agrupamento realizados usando o SEQ_CLUSTER, cujos resultados são avaliados e analisados via validação externa e dois índices de validação discutidos no Capítulo 4.

Capítulo 7: inicialmente resume os principais pontos levantados e investigados na pesquisa realizada, as conclusões derivadas dos experimentos conduzidos e, então, apresenta um conjunto de possíveis atividades que podem ser iniciadas, em continuidade ao trabalho desenvolvido e descrito nesta dissertação.

Capítulo 1. Aprendizado de Máquina: Principais Características e Conceitos Envolvidos

Este capítulo contextualiza a área de Aprendizado de Máquina (AM) na qual o projeto de pesquisa em nível de mestrado se insere, apresentando algumas de suas principais características e objetivos, bem como alguns conceitos fundamentais que subsidiam a área, com vistas a fornecer um embasamento à descrição da pesquisa realizada.

1.1 Aprendizado de Máquina

A área de Inteligência Artificial (IA) tem como um dos principais objetivos a proposta e a implementação de técnicas que viabilizam a incorporação de procedimentos considerados 'inteligentes', a sistemas computacionais. Apesar das muitas definições do que é inteligência, advindas das mais diferentes áreas do conhecimento humano, dois fatos devem ser considerados: (1) não existe consenso do que seja inteligência e (2) inteligência, independentemente de sua definição, envolve a capacidade de aprendizado. Inteligência está, pois, fortemente relacionada à capacidade de aprendizado exibida por 'aquilo' que é considerado 'ser inteligente', seja um ser humano, um animal ou um software.

A área de pesquisa de Aprendizado de Máquina (AM) é uma subárea da IA que, entre outros, investiga o desenvolvimento de formalismos e técnicas que permitem a construção de sistemas automáticos de aprendizado. O chamado *aprendizado indutivo de máquina* é o modelo de AM mais bem sucedido e o que mais tem sido implementado, utilizando inúmeras técnicas e algoritmos (ver por exemplo uma compilação dos algoritmos mais utilizados em (Mitchell 1997)). Uma maneira simplista de abordar aprendizado indutivo de máquina é como um processo com duas fases: (1) *treinamento*, na qual, a partir de um conjunto de situações concretas (referenciadas como *dados*, *instâncias* ou *exemplos*) que representam um conceito (tal conjunto é chamado *conjunto de treinamento*), uma descrição geral do conceito é aprendida – o processo de indução pode ser abordado como uma busca em um espaço de hipóteses, de

forma a encontrar aquela(s) que ‘melhor’ representa(m) os exemplos do conjunto de treinamento. Nesse contexto, ‘melhor’ pode ser definido em termos de certos critérios como, por exemplo, precisão e/ou compreensibilidade; (2) *classificação*, na qual a descrição geral do conceito aprendida na fase de treinamento é utilizada para a categorização de novos dados que são passados ao sistema.

Dentre as várias taxonomias propostas para organizar as muitas técnicas que implementam AM, aquela que adota como critério o nível da supervisão associado ao conjunto de treinamento, durante a fase de aprendizado, é de particular interesse para o desenvolvimento deste projeto. Essa taxonomia agrupa as técnicas em três diferentes grupos: (1) *aprendizado supervisionado*, (2) *aprendizado não-supervisionado* e (3) *aprendizado semissupervisionado*, que são caracterizados, de maneira simplificada, nos próximos parágrafos.

(1) Algoritmos de *aprendizado supervisionado* fazem uso de uma informação extra, chamada *classe* (ou categoria), que faz parte da descrição de cada dado de treinamento. A classe de cada dado é, geralmente, fornecida por uma fonte externa ao processo de aprendizado (por exemplo, por um especialista humano na área de conhecimento em questão). Dentre os algoritmos supervisionados mais bem sucedidos podem ser citados os identificados como (1) *simbólicos*, tais como: CN2 (Clark & Niblett 1989), ID3 (Quinlan 1986), C4.5 (Quinlan 1993), AQ (Michalski *et al*. 1983) e os (2) *neurais*, tais como: backpropagation (Bishop 1999), algoritmos neurais construtivos, tais como o Tower (Gallant 1990), Pyramid (Gallant 1993), BaBCoNN e MbabCoNN (Bertini & Nicoletti 2008, 2008a) etc., para problemas de classificação e o Cascade-Correlation (Fahlman & Lebiere 1991) para problemas de regressão (Nicoletti *et al*. 2009) (Bishop 1999).

(2) Algoritmos de *aprendizado não-supervisionado* não fazem uso da informação dada pela classe e, por essa razão, são tipicamente usados para a inferência do conceito a partir de dados cuja descrição não incorpora a classe do conceito que representam. Algoritmos não supervisionados geralmente aprendem por meio da identificação de subconjuntos de dados que compartilham certas similaridades. As várias famílias dos chamados algoritmos de agrupamento (brevemente abordadas no Capítulo 2) são representantes típicos deste grupo. Alguns tipos de redes neurais, como por exempo, as

de Hebb e de Kohonen, discutidas em detalhe em (Bishop 1999), também pertencem a esse grupo.

(3) Técnicas de *aprendizado semissupervisionado* são técnicas adequadas para situações nas quais o conjunto de treinamento é formado por dois subconjuntos: um conjunto (geralmente pequeno) constituído por dados que incorporam em suas descrições a informação da classe à qual pertencem e o outro (que é, via de regra, volumoso), por dados que não incorporam a informação da classe à qual pertencem. Algoritmos que implementam o aprendizado semissupervisionado geralmente utilizam o subconjunto de dados cuja descrição contém a classe para induzir uma expressão geral do conceito (usando algoritmos de aprendizado supervisionado) e, então, utilizam essa expressão para determinar a classe associada aos dados pertencentes ao outro conjunto. Dois representantes deste grupo de técnicas são os algoritmos Self-Training (Rosenberg *et al.* 2005) e o Co-Training (Blum e Mitchell 1998).

Dentre as inúmeras características que distinguem os métodos de aprendizado indutivo encontram-se (Nicoletti 1994):

(1) *Aprendizado incremental e não Incremental* – no caso incremental, a expressão do conceito vai sendo construída exemplo a exemplo e implica constante revisão por parte do algoritmo de AM; um novo dado pode, eventualmente, causar um rearranjo da expressão do conceito, para que este possa classificá-lo. A expressão do conceito vai se modificando à medida que os dados vão se tornando disponíveis. No caso não incremental, o conjunto de treinamento deve estar disponível desde o início do processo de aprendizado uma vez que a expressão do conceito é induzida considerando todos os dados de uma vez. Alguns exemplos de algoritmos de aprendizado incremental são: o ID4 (Schlimmer & Fisher 1986), o ID5 (Utgoff 1988) e o ID5R (Utgoff 1989) que lidam com a construção de árvores de decisão e são versões incrementais do conhecido algoritmo ID3 (Quinlan 1986). Além disso podem ser lembrados o Candidate-elimination (Mitchell 1982, 1997), que induz classificadores binários, o COBWEB (Fisher 1987), que induz taxonomias e categorizações em clusters, e o ILA (Giraud-Carrier & Martinez 1995) que induz classificadores representados por árvores binárias balanceadas. Os algoritmos não incrementais são bem mais numerosos e, dentre

eles, destacam-se: o ID3 (Quinlan 1986), o CN2 (Clark & Niblett 1989) e o backpropagation (Bishop 1999).

(2) *Aprendizado de um conceito e aprendizado de vários conceitos* – esta característica está relacionada à habilidade de um sistema poder aprender a expressão de apenas um ou, então, de vários conceitos de uma vez.

(3) *Uso (ou não) de Teoria do domínio* – se um sistema não tem informação a respeito do problema de aprendizado sendo abordado, supostamente deve induzir a expressão do conceito apenas a partir dos exemplos disponíveis. Para que soluções de problemas complexos de aprendizado sejam encontradas, entretanto, é fundamental que um volume substancial de conhecimento sobre o problema esteja disponível ao sistema de aprendizado, de maneira a subsidiar a indução do conceito. Esse conhecimento prévio existente é conhecido como teoria do domínio (ou conhecimento de fundo). Algoritmos de aprendizado automático caracterizados como Programação Lógica Indutiva (PLI) fazem uso substancial de teoria do domínio (e.g., FOIL (Quinlan 1990), GOLEM (Muggleton & Feng 1993), DUCE (Muggleton 1987), CIGOL (Muggleton & Buntine 1988), ITOU (Rouveirol 1992), CLINT (Raedt 1992) e MARVIN (Sammut & Banerji 1986)).

(4) *Linguagens de descrição de dados, de conceitos e de teoria do domínio* – em aprendizado indutivo os dados, teoria do domínio e as hipóteses formuladas são expressos em alguma linguagem e, geralmente, são usadas linguagens formais. Algoritmos tradicionais de aprendizado de máquina via de regra empregam linguagens proposicionais nas descrições de dados e de conceitos. Algoritmos que seguem a linha de Programação Lógica Indutiva, entretanto, empregam linguagens lógicas de primeira ordem às quais foram impostas restrições, para viabilizar o processo indutivo de aprendizado realizado pelo algoritmo.

(5) *Critérios de avaliação do conceito induzido* – entre os critérios mais usuais para se medir a qualidade do conceito induzido por um algoritmo de aprendizado estão:

(5.1) *Precisão de classificação* – geralmente medida como o percentual de exemplos corretamente classificados pela expressão (hipótese) induzida.

(5.2) *Transparência da descrição induzida* – em muitos domínios de aplicação (e.g. diagnóstico médico) é imprescindível que a descrição do conceito, gerada por um sistema de aprendizado, possa ser entendida por um ser humano. O entendimento não apenas aumenta a credibilidade no sistema de aprendizado, como também permite que o conceito possa ser assimilado e utilizado pelo especialista humano. Em muitas situações a transparência da descrição é medida pelo número de descritores e operadores usados na descrição do conceito.

(5.3) *Complexidade computacional* – está relacionada com os recursos computacionais necessários (tempo e espaço) para realizar o aprendizado.

1.2 Os Conjuntos de Treinamento, Teste e Validação em um Ambiente de Aprendizado de Máquina

Como brevemente mencionado na Seção 1.1, para viabilizar o aprendizado indutivo é imperativo que um conjunto de dados (também chamadas de exemplos ou instâncias), que representam os conceitos a serem aprendidos, esteja disponível. Tal conjunto é denominado *conjunto de treinamento*. Dados de treinamento são, geralmente, descritos por um conjunto de atributos cujos valores variam em um determinado intervalo e/ou conjunto de valores e, dependendo da situação, uma classe associada também participa da descrição (indicando qual conceito o dado em questão representa). A classe de cada dado que participa do conjunto de treinamento é, na maioria dos casos, determinada por um especialista humano da área de conhecimento descrita pelos dados.

O conjunto de treinamento é fornecido como entrada para o *software* que implementa um algoritmo de AM que, via de regra, devolve como saída expressões generalizadas da informação contida no conjunto de treinamento. A dependência do conjunto de treinamento para que o aprendizado possa acontecer, faz com que tais técnicas de aprendizado sejam caracterizadas como indutivas (em contraposição às técnicas de aprendizado dedutivas, muitas delas caracterizadas como “tradutores” de linguagens de representação de conhecimento baseadas em lógica).

A Figura 1.1 mostra um trecho de um dos muitos conjuntos de dados do conhecido repositório chamado *University of California at Irvine Machine Learning Repository* (*UCI Repository*) (Frank & Asuncion, 2010). O trecho em questão foi retirado do arquivo identificado como Iris, que contém dados numéricos de medidas que caracterizam flores íris pertencentes a três classes distintas: *iris setosa*, *iris virginica* e *iris versicolor*. O arquivo contem 150 dados; os 50 primeiros do arquivo descrevem flores iris da classe *iris setosa*, os 50 seguintes descrevem flores iris da classe *iris virginica* e, finalmente, os 50 últimos as da classe *iris versicolor*. O arquivo que contém esse conjunto apresenta os 150 dados agrupados por classe, na seguinte sequência: *setosa*, *virginica* e *versicolor*, respectivamente.

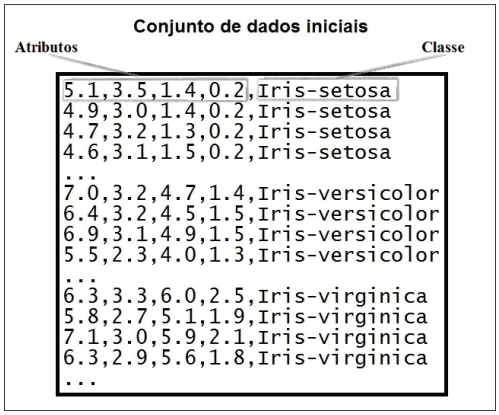


Figura 1.1 Trecho do conjunto de dados Iris (Frank & Asuncion, 2010), cujos dados são descritos por valores de quatro atributos numéricos (comprimento e largura da sépala e comprimento e largura da pétala), seguidos pela classe associada.

Cada dado é descrito por valores de quatro atributos numéricos relativos às medidas (em cm) de quatro características dessas flores, a saber: *comprimento da sépala*, *largura da sépala*, *comprimento da pétala*, *largura da pétala*, nessa ordem. Cada dado tem, também, ao final de sua descrição, a informação a qual *classe* de Iris as medidas anteriores se referem. É importante salientar que as 150 descrições não têm valores ausentes, problema comumente a ser encontrado em arquivos de dados reais e que deve ser tratado pelo algoritmo de aprendizado ou, então, por meio de um pré-processamento dos dados (como será abordado no Capítulo 4).

A Figura 1.2 exibe um esquema básico de AM, no qual um algoritmo de aprendizado (implementado como um *software*) induz um classificador (regra geral ou modelo) a partir de um conjunto de treinamento fornecido como entrada. Uma vez

induzido, o classificador pode então ser usado para classificar novos dados (de classe desconhecida).

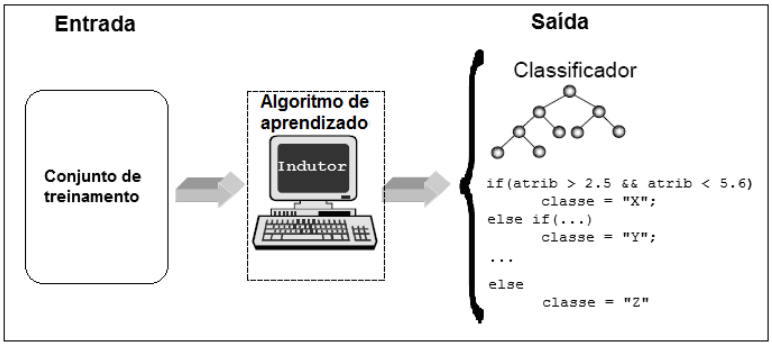


Figura 1.2 Esquema básico AM. Dado um conjunto de treinamento, o software (Indutor) que implementa o algoritmo de AM aprende (no caso), uma árvore de decisão que generaliza o conjunto de treinamento. Tal árvore pode ser facilmente traduzida em um conjunto de regras.

Em aprendizado supervisionado, cada dado de treinamento E_i que participa do conjunto de treinamento (Tr) é representado por um conjunto de valores de atributos e de uma classe associada. Em princípio todos os dados são descritos pelo mesmo conjunto de M atributos $\{A_1, A_2, ..., A_M\}$. Assim sendo, Tr = $\{E_1, E_2, ..., E_N\}$ tal que $E_i = (E_{i1}, E_{i2}, ..., E_{iM}, C_i)$, $E_{i,j}$ é um dos possíveis valores do correspondente atributo A_j (j=1, ..., M) para o dado E_i e C_i uma dentre as k possíveis classes i.e., $C_i \in \{C_1, C_2, ..., C_k\}$ ($1 \leq i \leq N$). É importante lembrar que cada dado de treinamento pertence a uma, dentre as várias classes (que devem ser mutuamente exclusivas). Como apontado em (Nicoletti *et al.*, 1998) em um sistema de aprendizado que faz uso de uma linguagem baseada em atributos para a representação de dados e conceitos, a tarefa de aprendizado (supervisionado) pode ser descrita como:

> *Dado um conjunto de exemplos de treinamento expressos como vetores de pares atributo-valor, cujas classes são conhecidas, encontrar uma regra que prediga a classe de um novo exemplo em função de seus atributos e valores.*

Com o objetivo de avaliar quão representativa é a expressão que foi induzida por um algoritmo de AM, um conjunto de dados, chamado *conjunto de teste*, é utilizado. Um conjunto de teste é, geralmente, um conjunto de dados independente do conjunto de treinamento mas que segue sua mesma distribuição de probabilidade. Se o classificador induzido expressa o conjunto de treinamento tão bem quanto o conjunto de teste, é

indicativo que um problema conhecido como *overfitting*, que pode ser considerado como um ajuste demasiado dos dados de treinamento, foi minimizado. Por outro lado, se o classificador expressa o conjunto de treinamento muito melhor do que o conjunto de teste, é sintoma que o problema de *overfitting* merece atenção.

De maneira simplista, um alto *overfitting* expressa a plasticidade da expressão do conceito de 'se acomodar' aos dados de treinamento; isso de certa forma torna o classificador inadequado para classificar novos dados i.e., dados que não participaram da indução do classificador. É importante lembrar que o objetivo de uma tarefa de aprendizado é, além de induzir um classificador baseado em dados, que o classificador induzido seja robusto (característica referenciada como a acurácia preditiva) quando usado com novos dados (diferentes daqueles que participaram de sua indução).

Com o objetivo de evitar *overfitting*, é prática recorrer a um terceiro conjunto de dados chamado *conjunto de validação*. Quando se busca um classificador mais adequado para uma determinada aplicação, o conjunto de treinamento é usado como entrada para diferentes algoritmos de AM e cada um deles irá induzir um classificador; o conjunto de teste é então usado para comparar os desempenhos desses classificadores, com o objetivo de identificar o melhor deles. Conjuntos de validação são geralmente empregados para inferir características de desempenho, tais como precisão, sensitividade e especificidade, etc.

A Figura 1.3 exemplifica uma situação simples na qual um conjunto pequeno de dados é dividido em dois subconjuntos disjuntos, o de treinamento, com 75% dos dados e o de teste, com os restantes 25%. A divisão do conjunto original de dados em conjunto de treinamento e conjunto de teste pode ser realizada de forma aleatória, de modo a garantir que os dois conjuntos sejam amostras aleatórias da mesma distribuição.

Como comentado em (Keller 2012), às vezes o conjunto de validação é importante para ajustar o classificador e evitar possíveis falhas. É o caso, por exemplo, de uma divisão em treinamento e teste ter sido enviezada, e dados de uma determinada classe não se encontram presente no conjunto de treinamento.

Osuna em (Osuna 2012) faz uma revisão de várias técnicas de validação voltadas para dois problemas fundamentais em reconhecimento de padrões: seleção de modelos e estimativa de desempenho.

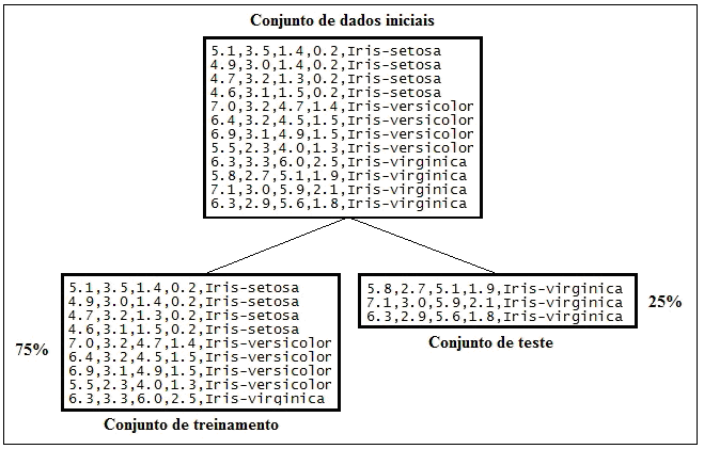


Figura 1.3 Conjunto de dados iniciais dividido em conjunto de treinamento e conjunto de teste.

A técnica de validação conhecida como *cross-validation* (*validação cruzada*) é um processo adotado pela comunidade de AM para a indução de classificadores potencialmente eficientes. Como comentado em (Keller, 2012) e (Arlot & Celisse 2010), tal técnica pode ser considerada um processo de reamostragem dos dados, com o objetivo de reduzir falhas e que, basicamente, consiste na repetição sistemática de vários treinamentos e testes parciais.

Na validação cruzada os dados disponíveis são divididos aleatoriamente em *k folds* (ou subconjuntos) com aproximadamente o mesmo número de dados. O processo de aprendizado é então realizado tendo como entrada dados pertencentes a $k–1$ subconjuntos; o subconjunto restante é então usado como conjunto de teste. O processo é sistematicamente repetido k vezes; dessa forma, cada um dos k subconjuntos é usado uma vez como conjunto de teste.

A Figura 1.4 mostra um exemplo do processo de 5-validação cruzada, a partir de um conjunto de dados contendo 15 dados. Na figura o conjunto de treinamento (formado por $k–1 = 4$ subconjuntos) é indicado por “Tr” e o conjunto de teste por “Te”.

Mesmo com várias divisões, entretanto, a partição aleatória não garante que cada dado aparecerá em pelo menos um conjunto de teste. A técnica chamada *leave-one-out* (*deixe-um-fora*) pode ser tratada como uma variante da validação cruzada em que $k = n$, onde n = número de dados do conjunto original e que garante que todos os dados serão (um por vez), tratados como dado de teste. O processo consiste em n repetições de: (1)

indução do classificador usando *n–1* dados seguida de (2) avaliação do classificador induzido no passo (1) usando o único dado que não participou do processo indutivo. O erro final é a média dos erros calculados em cada uma das n repetições do processo.

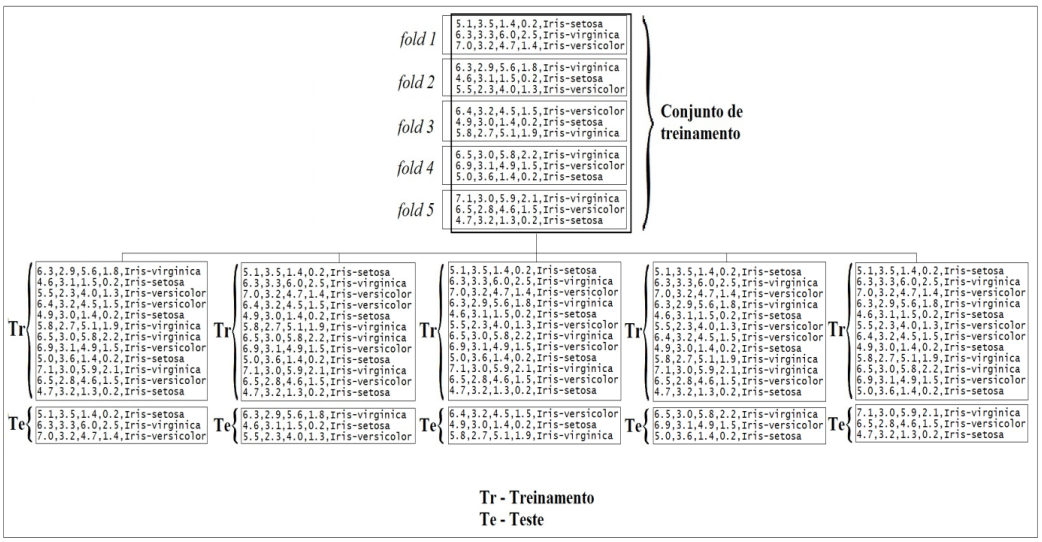


Figura 1.4 Esquema do processo de *k*-validação cruzada (*k=5*).

A Figura 1.5 mostra um exemplo do processo deixe-um-fora. Como o conjunto original tem 12 dados, são realizadas 12 repetições do seguinte processo de dois passos: (1) indução do classificador usando como conjunto de treinamento 11 dados e (2) avaliação do classificador no dado restante que não participou do processo indutivo.

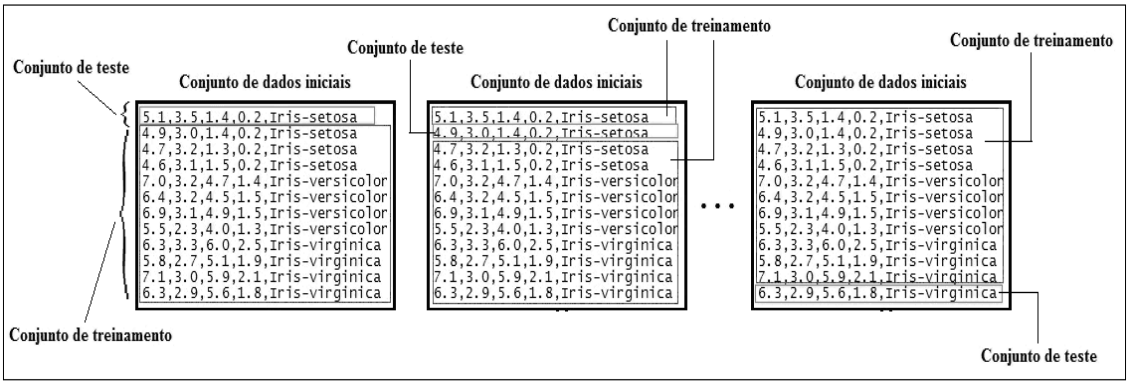


Figura 1.5 Esquema do processo *Leave-one-out*.

1.3 Os Vários Tipos de Atributos que Descrevem os Dados

Como visto na Seção 1.2, em aprendizado supervisionado cada dado E_i que participa do conjunto de treinamento (Tr) é representado por um conjunto de valores de atributos e de uma classe associada. Em princípio todos os dados são descritos pelo mesmo conjunto de M atributos $\{A_1, A_2, ..., A_M\}$. Assim sendo, $Tr = \{E_1, E_2, ..., E_N\}$ tal que $E_i = (E_{i1}, E_{i2}, ..., E_{iM}, C_i)$, sendo E_{ij} um dos possíveis valores do correspondente atributo A_j (j=1, ..., M) e C_i uma dentre as k possíveis classes i.e., $C_i \in \{C_1, C_2, ..., C_k\}$ $(1 \leq i \leq N)$. Atributos que descrevem os dados podem assumir valores em intervalos contínuos ou em um conjunto finito discreto. De acordo com (Kohavi & Provost 1998) os domínios mais comumente usados para representar valores de atributos são aqueles identificados como:

(1) categórico (nominal ou ordinal): caracterizado por um número finito de valores discretos; o subtipo nominal expressa a inexistência de ordem entre os valores considerados enquanto que o ordinal expressa a existência de uma relação de uma ordem entre tais valores;

(2) contínuo (quantitativo): caracterizado como a representação de números reais (lembrando sempre que, a representação real em computadores é, na verdade, discreta).

A caracterização do valor associado à classe pode ser discreta ou real. Tal distinção estabelece uma nítida divisão do conjunto de algoritmos supervisionados de AM; de um lado estão os algoritmos caracterizados como classificadores (classes com valores extraído de um conjunto discreto de valores) e de outro, os algoritmos caracterizados como de regressão (classes com valores reais). Dados de treinamento utilizados em algoritmos de aprendizado não supervisionados, como aqueles usados em agrupamentos, são também descritos por atributos com valores categóricos ou contínuos, como os utilizados em algoritmos de aprendizado supervisionado; neles, entretanto, o atributo classe não comparece. No que segue a mesma notação adotada em aprendizado supervisionado, como descrita no primeiro parágrafo desta seção, será também adotada; obviamente, como se trata de aprendizado não supervisionado, a classe à qual um dado pertence não comparece na sua descrição e para diferenciar, o conjunto inicial de pontos de dados que será agrupado será referenciado como CP.

Capítulo 2. Aprendizado Não-supervisionado e Algoritmos de Agrupamento

Este capítulo caracteriza aprendizado de máquina não-supervisionado e discute alguns dos aspectos mais relevantes relacionados a algoritmos que implementam aprendizado não-supervisionado conhecidos como *algoritmos de agrupamento*. O objetivo é contextualizar essa subárea específica de AM e fornecer subsídios teóricos para o Capítulo 3, no qual um conjunto de algoritmos não-supervisionados, objeto da pesquisa descrita nesta dissertação, é abordado.

2.1 Aprendizado Não-supervisionado

Como discutido no Capítulo 1, a classe de cada dado que participa do conjunto de treinamento é, na maioria dos casos, determinada por um especialista humano da área de conhecimento descrita pelos dados. O fato da classe participar da descrição do dado e do método de aprendizado fazer uso dele, caracteriza a técnica como *aprendizado supervisionado* (a disponibilidade da classe associada a cada dado de treinamento é considerada supervisão externa).

Em muitas situações do mundo real, entretanto, a classe à qual cada dado pertence pode ser: (1) desconhecida; (2) não existir um especialista humano com conhecimento suficiente que seja capaz de, com base na descrição dos valores de seus atributos, estabelecer a classe do dado ou (3) a determinação da classe é um processo caro que envolve, por exemplo, testes de laboratório ou a contratação de um grupo de especialistas para sua definição.

Métodos de aprendizado caracterizados como não-supervisionados, apesar de também requererem um conjunto de treinamento, pressupõem que a classe de cada dado de treinamento não está incorporada à sua descrição (contrário ao que acontece com métodos supervisionados, para os quais a informação da classe de cada dado é fundamental para a indução apropriada do conceito). Devido à indisponibilidade da classe associada a cada dado, tais métodos não fazem uso dessa informação. A estratégia de aprendizado utilizada por métodos não-supervisionados é a de tentar evidenciar uma organização dos dados em grupos apropriados, o que permitiria

descobrir similaridades e diferenças entre os dados e, então, derivar conclusões sobre eles.

Como apontado em (Theodoridis & Koutroumbas 2009), técnicas de agrupamentos são utilizadas nas mais variadas áreas de conhecimento e, muitas vezes, sob diferentes nomes. Na área de reconhecimento de padrões, por exemplo é chamada de aprendizado não-supervisionado e, também, aprendizado sem um tutor; em biologia e ecologia, é conhecida como taxonomia numérica; em ciências sociais, como tipologia e em teoria dos grafos como partição.

2.2 Considerações Sobre um Problema de Agrupamento

Esta subseção inicialmente contempla um exemplo inspirado naquele apresentado em (Theodoridis & Katoumbra 2009), que permite uma caracterização simples de um procedimento de agrupamento e permite estabelecer várias observações sobre o método, de maneira a fornecer subídios para, na subseção seguinte, abordar os vários passos que devem ser considerados quando da proposta/implementação de um algoritmo de agrupamento.

2.2.1 Organizando um Conjunto de Dados – um Exemplo de Agrupamento

Considere a Tabela 2.1 que lista um conjunto de animais e um conjunto de características que os descrevem. Cada linha da Tabela 2.1 é considerado um dado (ou instância de dado); a primeira coluna, entretanto, pode ser considerada como um identificador de cada dado. Note que o valor atribuído a cada uma das seis características que descrevem o animal depende de do tipo de animal à que se referem. Características podem ser usadas para agrupar tais animais, organizando-os.

Se o critério de organização for definido, por exemplo, apenas pela característica *mamam*, os 12 animais são divididos em dois grupos i.e., aqueles que mamam, G_1 = {baleia, cabra, cachorro, ornitorrinco, rato} e aqueles que não o fazem i.e. o grupo G_2 = {andorinha, avestruz, cascavel, jacaré, pardal, sapo, sardinha}. Um agrupamento organizado utilizando apenas tal característica é então dado por: {G_1, G_2} = {{baleia, cabra, cachorro, ornitorrinco, rato},{andorinha, avestruz, cascavel, jacaré, pardal, sapo, sardinha}}. Já se o critério for *vivem na água*, o conjunto de 12 animais é particionado

em três grupos, a saber, G_1 = {baleia, sardinha}, G_2 = {jacaré, ornitorrinco, sapo} e G3 = {andorinha, avestruz, cabra, cachorro, cascavel, pardal, rato} e o agrupamento que definem é {G_1, G_2, G_3} = {{baleia, sardinha}, {jacaré, ornitorrinco, sapo},{andorinha, avestruz, cabra, cachorro, cascavel, pardal, rato}}. O conjunto de animais que satisfazem ao critério composto por ambas características, i.e., *voam* e *mamam*, é vazio. O conjunto de animais que satisfazem ao critério composto pelas características *mamam* e *botam ovos* é um conjunto unitário: {ornitorrinco}. Note também que se a identificação do animal for tratada como uma característica, os doze animas vão estar organizados em 12 grupos unitários.

Tabela 2.1 Conjunto de animais e de algumas de suas características.S (Sim), S/N (Sim e Não) e em branco: Não

animais	têm penas	têm bicos	mamam	voam	vivem na água	botam ovos
andorinha	S	S		S		S
avestruz	S	S				S
baleia			S		S	
cabra			S			
cachorro			S			
cascavel						S
jacaré					S/N	S
ornitorrinco		S	S		S/N	S
pardal	S			S		S
rato			S			
sapo					S/N	S
sardinha					S	S

A escolha do critério usado para organizar o conjunto inicial de dados é de fundamental importância para obter uma organização racional dos dados em grupos e é fortemente dependente da especificação do problema. O exemplo descrito pelos dados da Tabela 2.1 mostra que o processo de agrupamento de animais pode levar a resultados diferentes, dependendo do(s) critério(s) definido(s). Como comentado em (Tan *et al.* 2005), grupos conceitualmente significativos de objetos (compartilhando características) desempenham um papel relevante em como seres humanos descrevem e analisam o mundo. Seres humanos são habilidosos em dividir um conjunto de objetos em grupos (agrupamento) e também em atribuir determinados objetos a esses grupos

(classificação). Em um contexto de entendimento de dados, grupos são potenciais classes.

Alguns algoritmos de agrupamento caracterizam cada grupo de dados que participa do agrupamento por meio de um único elemento, o *protótipo* (ou *representante*, ou ainda *representativo*) i.e., aquele dado que representa os dados do grupo. Note que com esse tipo de representação, ao invés do grupo ser representado por todos os dados que o definem (dependendo do domínio podem existir milhares), é representado por apenas um dado, o que traz vantagens em termos de armazenamento e acesso. Em domínios de dados descritos por atributos com valores contínuos (i.e., número reais) o protótipo de um grupo é, geralmente, o *centróide* do grupo i.e., a média de todos os pontos pertencentes ao grupo em questão. Em dados descritos por atributos que têm valores categóricos (por exemplo aqueles da Tabela 2.1), o protótipo é via de regra o *medóide* i.e., o ponto mais representativo do grupo. Conceitualmente medóides e centróides são similares; entretanto, o medóide de um grupo é sempre um elemento do grupo que está representando. Medóides são comumente usados quando o contróide não pode ser definido.

Considere um conjunto inicial de 17 pontos bidimensionais como o mostrado na Figura 2.1 e suponha que tais pontos (•) tenham sido organizados em três grupos distintos A = {(1 1), (0,5 2), (1,5 2), (2 1,5)}, B = { (4 4), (4 5), (4,5 4,5), (5 4), (5 5)} e C = { {(7 1), (7 2), (7,5 2,5), (8 1), (8 3), (8,5 1,5), (9 2), (9 3)}como mostra a figura. Nela o símbolo × representa a posição do centróide associado a cada um dos grupos (ver Tabela 2.2). Note que, particularmente no grupo B o centróide coincide com um dos pontos do grupo enquanto que nos outros dois grupos os centróides não coincidem com nenhum dos pontos dos respectivos grupos que representam.

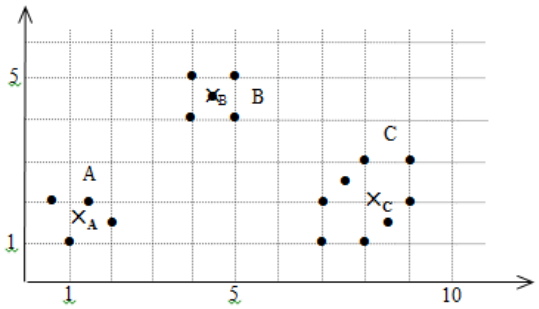


Figura 2.1 Um conjunto com 17 dados bi-dimensionais (•), agrupados em 3 grupos (A, B e C) e os respectivos centróides de cada grupo i.e., $\times_A$, $\times_B$, $\times_C$.

Tabela 2.2 Cálculo dos centróides dos grupos de dados do agrupamento da Figura 2.1.

	$\times_A$	$\times_B$	$\times_C$
abcissa	(1+0,5+1,5+2)/4 = 5/4 = 1,25	(4+4+5+5+4,5)/5 = 22,5/5 = 4,5	(7+7+7,5+8+8+8,5+9+9)/8=64/8=9,25
ordenada	(1+2+2+1,5)/4 = 6,5/4 = 1,625	(4+5+4+5+4,5)/5 = 22,5/5 = 4,5	(1+2+2,5+1+3+1,5+2+3)/8=16/8=2

2.2.2 Considerações para a Implementação de um Processo de Agrupamento

Os autores em (Jain *et al.* 1999) descrevem um conjunto de especificações que devem ser consideradas e decisões que devem ser tomadas, quando da implementação de um procedimento de agrupamento, resumidas a seguir.

(1) *Representação dos dados* - como são descritos, quais tipos de atributos são usados para descrevê-los (numéricos (inteiros ou reais?), categóricos, etc.) e qual a escalabilidade dos atributos que descrevem os dados. Opcionalmente, pode ser considerado se tais dados devem ser pré-processados com vistas à seleção de atributos relevantes;

(2) *Medida de proximidade* - o conceito de proximidade é, geralmente, definido por meio da escolha de uma função de distância entre pares de dados; espera-se que a escolha seja apropriada ao domínio de dados em questão. Existe uma grande variedade de tais funções (ver, por exemplo (Anderberg 1973; Jain & Dubes 1988; Diday & Simon 1976)). Uma medida de distância simples que é frequentemente utilizada é a distância Euclidiana.

(3) *Tarefa de agrupamento* - pode ser realizada de diferentes maneiras. O agrupamento (ou agrupamentos resultantes), dependendo do algoritmo adotado, pode ser *crisp* (cada um dos dados pertence a um único grupo do agrupamento) ou *fuzzy* (um dado pode pertencer a vários grupos, com graus de pertinência diferenciados). Algoritmos de agrupamento hierárquicos produzem uma série de partições aninhadas, usando um critério para fusão ou separação de grupos, subsidiado por similaridade. Já algoritmos de agrupamento identificados como particionais identificam uma partição dos dados que otimiza (localmente, na maioria dos casos) o critério de agrupamento.

(4) *Abstração dos dados* (se necessário) - processo de extração de uma representação compacta e simples do agrupamento gerado. Simplicidade pode ser abordada (a) sob a perspectiva da análise automática (de maneira que um *software* possa utilizar o

agrupamento obtido em tarefas subsequentes), ou (b) orientada à seres humanos, de maneira que a representação possa ser facilmente entendida e assimilada por usuários de um determinado sistema. No contexto de agrupamentos, uma típica abstração é a descrição compacta de cada um dos grupos do agrupamento usando seu representativo (como visto na Seção 2.2.1).

(5) *Avaliação dos resultados obtidos* (se necessário) - diz respeito ao uso de mecanismos para avaliar a qualidade do agrupamento obtido (Dubes 1993) e, geralmente, implementada por meio de medidas estatísticas (como será discutido em detalhes no Capítulo 4).

Theodoridis & Koutroumbas em (Theodoridis & Koutroumbas 2009) também sugerem considerações básicas com vista à implementação de procedimento de agrupamento em conjuntos de dados e que são resumidas a seguir. Note que não diferem em muito daquelas sugeridas em (Jain & Dubes 1988) apresentadas anteriormente.

(1) *Seleção de atributos*: considerando que os dados estão disponibilizados como vetores de valores de atributos (como padronizado na Seção 1.2), esse passo busca evidenciar aqueles atributos que efetivamente são relevantes na caracterização de grupos de dados. Dados que representam registros médicos de pacientes referentes a exames realizados, por exemplo, o atributo que informa o endereço do paciente é irrelevante para um processo que busca agrupar registros de pacientes com base no(s) resultado(s) de exame de sangue (exame de TSH, por exemplo).

(2) *Medida de proximidade ou similaridade*: quantifica o grau de semelhança entre dois dados descritos pelo mesmo conjunto de atributos; tal medida é obtida, geralmente, por um cálculo da distância entre os dados.

(3) *Critério de agrupamento*: é dependente do tipo de grupo ‘esperado’, de acordo com a intuição do especialista sobre o tipo de grupo que melhor modela os dados a serem agrupados. Grupos de dados compactos, por exemplo, podem ser sensíveis a um determinado critério enquanto grupos alongados de dados, sensíveis a outro. O processo de agrupamento pode ser hierárquico, com um processo recursivo de junções ou separações de grupos, ou não-hierárquico, com o emprego direto de técnicas de

discriminação de grupos, tais como as usadas em algoritmos sequenciais de agrupamento (Capítulo 3);

(4) *Algoritmo de agrupamento*: após adotada uma medida de proximidade e um critério de agrupamento, é necessário definir/escolher um algoritmo de agrupamento a ser usado. Diferentes algoritmos podem apresentar comportamentos diferentes e, também, resultados diferentes, para um mesmo conjunto de dados;

(5) *Validação dos resultados*: o(s) agrupamento(s) gerado(s) por um algoritmo de agrupamento precisam, então, ser validados, com o objetivo de ratificar sua corretude. Via de regra o processo de validação é implementado por meio de medidas estatísticas que permitem estimar a corretude do(s) agrupamento(s) resultante(s);

(6) *Interpretação dos resultados*: é comum a necessidade de integração dos resultados de um procedimento de agrupamento a (a) outras técnicas, (b) evidências experimentais, (c) análises experimentais, com vistas a promover a robustez do processo de inferência de conclusões - agrupamento passa então a ser uma parte de todo um sistema computacional disponibilizado.

Como pode ser inferido dos dois conjuntos de especificações apresentados, a implementação de um procedimento de agrupamento não é um processo simples que pode ser desenvolvido de maneira trivial. Tal processo envolve um número razoável de considerações e decisões que, para ser bem sucedido, deve estar subsidiado por um profundo conhecimento da área de aplicação da qual provêm os dados a serem agrupados, bem como de conhecimento empírico que um especialista humano tem sobre os próprios dados.

2.3 Taxonomia de Algoritmos de Agrupamento

Na literatura podem ser encontrados inúmeros algoritmos de agrupamento, subsidiados pelos mais variados formalismos matemáticos e estatísticos. Uma maneira de organizá-los com vistas ao estudo sistemático desses algoritmos é por meio de uma taxonomia. No que segue três tipos que estão disponibilizados na literatura são apresentadas com intuito de evidenciar perspectivas distintas que norteiam propostas de taxonomias de algoritmos de agrupamento. Theodoridis & Koutroumbas em

(Theodoridis & Koutroumbas 2009) organizam tais algoritmos nas seguintes categorias principais:

(1) *Algoritmos sequenciais*: generalmente caracterizados como simples e rápidos, produzem como resultado um único agrupamento. Os dados a serem agrupados podem ser apresentados aos algoritmos uma ou algumas vezes e, via de regra, o resultado final depende da ordem em que tais dados são apresentados. Algoritmos caracterizados como sequenciais tendem a gerar agrupamentos compactos com formas esféricas ou elipsóidais, na dependência da medida de distância usada. Algoritmos sequenciais compartilham algumas características, tais como: necessidade de um ou poucos passos, o número de grupos não é conhecido inicialmente e, geralmente, têm como entrada um limiar e o número máximo de grupos a serem criados. Os grupos são definidos por meio de um cálculo de distância apropriado entre um dado e um agrupamento, levando em consideração o limiar associado a essa distância. Exemplos de algoritmos sequenciais: *Basic Sequential Algorithmic Scheme* (BSAS), *Modified Basic Sequential Algorithmic Scheme* (MBSAS) e *Two-Threshold Sequencial Algorithmic Scheme* (TTSAS); esses três algoritmos são objeto de estudo e pesquisa deste projeto de mestrado e são abordados com mais detalhe no Capítulo 3.

(2) *Algoritmos hieráquicos*: produzem uma sequência de grupos de dados aninhados, resultado de partições sucessivas dos dados, e promovem a representação hierárquica dos dados de entrada. Algoritmos hierárquicos podem ser subdivididos em duas subcategorias, aglomerativos e divisivos:

> (2.1) *Algoritmos aglomerativos*: essa subcategoria de algoritmos produz uma sequência de agrupamentos com um número decrescente de grupos. O agrupamento produzido em um passo *p* é baseado no agrupamento produzido no passo *p*-1, no qual dois grupos são unidos, diminuindo assim o número de grupos a cada passo.
>
> (2.2) *Algoritmos divisivos*: funcionam de maneira oposta aos aglomerativos. Produzem uma sequência de agrupamentos cujo número de grupos aumenta a cada passo. O agrupamento produzido em um passo *p* é resultante da divisão de um único grupo (do agrupamento obtido no passo *p*-1), em dois grupos.

(3) *Algoritmos baseados em otimização da função de custo*: essa categoria agrupa algoritmos que são dependentes de uma função de custo *J*, usada para avaliar o agrupamento. Tais algoritmos usam conceitos do cálculo diferencial e produzem agrupamentos sucessivos na tentativa de otimizar *J*. Usam como critério de parada, geralmente, a determinação de um ótimo local. Entre as subcategorias englobadas por algoritmos que usam uma função de custo estão as três a seguir; detalhes sobre essas e outras subcategorias podem ser encontrados na mesma referência:

(3.1) algoritmos *hard (*ou *crisp)*;

(3.2) algoritmos probabilísticos;

(3.3) algoritmos nebulosos (*fuzzy*).

(4) *Outros modelos*: agrupam algoritmos que não pertencem às categorias anteriores, como por exemplo, algoritmos de agrupamento que usam algoritmos genéticos, algoritmos baseados na teoria dos grafos, algoritmos de agrupamento por sub-espaço, algoritmos de agrupamento baseados em operadores de morfologia binária, entre outros.

2.4 Conceitos e Definições Relevantes Associados a Agrupamentos

A técnica de agrupamento, cujo resultado é também chamado agrupamento, é um procedimento que busca particionar um conjunto de dados (objetos) em grupos usando, para isso, algum critério que, via de regra, é fundamentado nos próprios dados (i.e., sua descrição).

O mecanismo básico que um procedimento de agrupamento (também identificado como algoritmo de agrupamento) implementa é o de comparar dados entre si e agrupá-los convenientemente em grupos. O conjunto de todos os grupos criados é o que se chama de agrupamento.

Os autores em (Jain *et al.* 1999) definem informalmente agrupamento como a organização de uma coleção de dados (geralmente representados como vetores de medidas ou, então, um ponto em um espaço multidimensional) em grupos com base na similaridade. Intuitivamente, dados que pertencem a um mesmo grupo são mais semelhantes entre si do que dados que pertencem a grupos distintos.

Seja CP = $\{E_1, E_2, ..., E_N\}$ um conjunto contendo N dados M-dimensionais, i.e., cada um deles descrito por M atributos i.e., $A_1, A_2, ..., A_M$. Um K-agrupamento de CP pode ser definido com uma partição de CP em K conjuntos (grupos), $G_1, G_2, ..., G_K$. Se um agrupamento dos dados do conjunto CP é definido como uma partição de CP, então as seguintes três condições devem ser verificadas:

(1) $G_i \neq \emptyset$, i = 1, ..., K

(2) $\bigcup_{i=1}^{K} G_i = CP$

(3) $G_i \cap G_j = \emptyset$, $i \neq j$ e i, j = 1, ..., K

Assume-se que os dados agrupados em G_i (i = 1, ..., K) sejam "mais semelhantes" entre si do que dados que pertencem a grupos.

2.5 Medidas de Similaridade e Distâncias

Uma vez que o conceito de similaridade é parte integrante de um processo de agrupamento, a definição de uma medida de similaridade entre dois pontos de dados, extraídos de um mesmo espaço de atributos, é essencial a qualquer procedimento de agrupamento. Na maioria dos casos a definição da medida de similaridade é determinante para a indução de um agrupamento que seja representativo e que espelhe a real natureza da organização dos dados.

Como aconselhado em (Jain *et al.* 1999), devido à diversidade de tipos de atributos e de suas respectivas unidades de medida, a medida de distância deve ser cuidadosamente escolhida. Via de regra é mais comum calcular a *dissimilaridade* entre dois pontos de dados usando uma medida de distância definida no espaço de atributos.

No que segue, o foco será em medidas de distância usada para pontos de dados descritos por atributos com valores contínuos - a métrica mais popular para atributos contínuos é a *distância Euclidiana*. Ainda em (Jain *et al.* 1999), é lembrado que a distância Euclidiana é adequada quando o conjunto de dados tem grupos 'compactos' ou 'isolados', como também comentado em (Mao & Jain 1996). A inconveniência de usar diretamente métricas de Minkowski é causada pela tendência de atributos com valores maiores dominarem os demais. Soluções para esse problema incluem a normalização dos atributos contínuos (a um escopo ou variância comum) ou esquemas

que envolvem ponderação. Tal métrica é a mais utilizada quando os atributos possuem valores contínuos para avaliar a proximidade de dados representados em duas ou três dimensões. A distância de Minkowski é dada pela Equação (2.1), onde *M* é o número de atributos do dado. A variação do parâmtro *p* define distâncias diferentes; quando *p* = 2 é calculada a distância Euclidiana.

$$dist(E_x, E_y) = \sqrt[p]{\sum_{i=1}^{M} (| E_{x_i} - E_{y_i} |)^p}, p \geq 1 \qquad (2.1)$$

Correlação linear entre atributos pode também distorcer medidas de distância; tal distorção pode ser minimizada por meio da aplicação de uma transformação aos dados ou pelo uso da distância de Mahalanobis ao quadrado, que é uma distância estatística de um dado E_x e um centróide E_y dada pela Equação (2.2), onde *S* representa a matriz de covariância.

$$dist(E_x) = \sqrt{(E_x - E_y)^T S^{-1} (E_x - E_y)} \qquad (2.2)$$

Como discutido em (Jain *et al.*, 1999), a determinação de distâncias entre pontos de dados em que alguns ou todos os atributos são discretos é problemática uma vez que tipos diferentes de atributos não são comparáveis e (como um exemplo extremo) a noção de proximidade é efetivalemente bi-valorada para atributos nominais. Discussões sobre uma vasta variedade de outras métricas podem ser encontradas em (Diday & Simon 1976) e (Ichino & Yaguchi 1994).

No que segue são apresentados alguns conceitos e fórmulas relevantes aos assuntos tratados nos próximos capítulos.

Considere dois pontos de dados E_x e E_y pertencentes a um espaço M-dimensional, notados respectivamente por $E_x = (E_{x_1}, E_{x_2}, ..., E_{x_M})$ e $E_y = (E_{y_1}, E_{y_2}, ..., E_{y_M})$.

(a) a distância Euclidiana (*dist*) entre dois pontos de dados E_x e E_y (ou, alternativamente, entre E_y e E_x) é dada pela Equação (2.3). A distância Euclidiana entre os pontos E_x e E_y representa o comprimento do segmento de reta que os conecta.

$$\mathrm{dist}(E_x, E_y) = \sqrt[2]{(E_{x_1} - E_{y_1})^2 + (E_{x_2} - E_{y_2})^2 + \cdots + (E_{x_M} - E_{y_M})^2} \qquad (2.3)$$

(b) um ponto em um espaço euclidiano M-dimensional pode ser abordado como um vetor euclidiano. Assim sendo, os pontos E_x e E_y podem ser vistos como vetores euclidianos, cujas origens coincidem com a origem do espaço e cujos fins coincidem com os pontos E_x e E_y, respectivamente. A *norma euclidiana* (ou *comprimento euclidiano* ou ainda *magnitude*) de um vetor E_x (notada por $\|E_x\|$) mede o comprimento do vetor e é dada pela Equação (2.4).

$$\|E_x\| = \sqrt[2]{{E_{x_1}}^2 + {E_{x_2}}^2 + \ldots + {E_{x_M}}^2} \qquad (2.4)$$

Note que a norma euclidiana de um vetor E_x é dada pela raiz quadrada do *produto escalar* (**.**) (ou produto interno, no contexto de um espaço euclidiano) de E_x pelo vetor transposto E_x^T como estabelece a Equação (2.5).

$$\|E_x\| = (E_x^T . E_x)^{\frac{1}{2}} \qquad (2.5)$$

A distância euclidiana padrão entre dois pontos E_x e E_y pode ser elevada ao quadrado com o objetivo de colocar, progressivamente, maior peso em pontos que estão mais distantes. Nesse caso a Equação (2.3) é substituída pela Equação (2.6). Note entretanto que a distância euclidiana ao quadrado não é uma métrica, uma vez que não satisfaz à desigualdade triangular; é contudo frequentemente usada em problemas de otimização em que distâncias apenas têm que ser comparadas (ver http://en.wikipedia.org/wiki/Euclidean_distance).

$$\mathrm{dist}(E_x, E_y)^2 = (E_{x_1} - E_{y_1})^2 + (E_{x_2} - E_{y_2})^2 + \ldots + (E_{x_M} - E_{y_M})^2 \qquad (2.6)$$

O produto escalar é uma operação algébrica que opera sobre duas sequências numéricas com o mesmo número de elementos (geralmente coordenadas de vetores) e retorna um único número. Algebricamente definido, o produto escalar é a soma dos produtos das correspondentes posições nas duas sequências numéricas.

Geometricamente, é o produto das magnitudes dos dois vetores pelo coseno do ângulo formado entre eles.

2.6 Considerações Finais

O que se buscou apresentar nesse capítulo foram os principais conceitos e definições associados à algoritmos de agrupamento. Devido ao extenso volume de publicações e pesquisa na área, a composição do capítulo se restringiu ao mínimo necessário para cumprir o seu único objetivo, que é o apresentar as principais ideias, conceitos e estabelecer notação necessária relativos a agrupamentos, para subsidiar os capítulos que seguem.

Capítulo 3. A Família de Algoritmos Sequenciais de Agrupamento

Este capítulo apresenta e discute uma família de algoritmos chamada *Basic Sequential Algorithmic Scheme* (BSAS) que tem por base um algoritmo de agrupamento de mesmo nome e dois outros dele derivados. O principal objetivo do capítulo é apresentar as principais características dos algoritmos dessa família bem como discutir as motivações que subsidiaram suas propostas. Inicialmente, entretanto, o capítulo apresenta na Seção 3.1 o algoritmo de agrupamento conhecido como K-Means, que será utilizado como algoritmo base para comparação com os outros algoritmos descritos neste capítulo. A Seção 3.2 aborda o algoritmo BSAS. Em seguida, as seções 3.3 e 3.4 abordam dois refinamentos do BSAS conhecidos respectivamente como MBSAS e TTSAS. Na Seção 3.5 são apresentadas duas estratégias de refinamento pós-agrupamento. Na Seção 3.6 são comentados e discutidos alguns trabalhos de pesquisa identificados e superficialmente analisados no Exame de Qualificação, que foram investigados com mais detalhes com vistas a possíveis contribuições ao desenvolvimento do trabalho de pesquisa de mestrado.

3.1 O Algoritmo de Agrupamento K-Means (K-Médias)

O K-Means (também conhecido como C-Means ou Isodata) é, sem sombra de dúvida, o algoritmo de agrupamento mais conhecido e popular dentre todos os algoritmos de agrupamento. O K-Means pode ser descrito de uma maneira simplista como um algoritmo particionador cujo objetivo é encontrar uma partição do conjunto de dados de entrada em K grupos disjuntos. Cada grupo é representado pelo centroide do grupo (ver Seção 2.2.1) que, via de regra, é definido como o ponto médio dos pontos de dados que pertencem ao grupo.

O pseudocódigo do K-Means é descrito em Algoritmo 3.1. Primeiramente, K pontos são aleatoriamente escolhidos (os centroides iniciais dos K grupos); o valor do parâmetro K é fornecido pelo usuário e indica a quantidade desejada de grupos no agrupamento. Todo ponto do conjunto de dados que é entrada para o algoritmo é, então, associado àquele centróide do qual se encontrar mais próximo, e cada conjunto de pontos associado a um centróide constitui um grupo do agrupamento. O centróide de

cada grupo de pontos é então atualizado, de maneira a refletir a média dos pontos que pertencem ao grupo. O processo se repete até nenhum ponto mudar de grupo, como descreve o Algoritmo 3.1.

O K-Means é um algoritmo simples e eficiente que, como pode ser confirmado na literatura, tem sido adaptado para ser usado em muitos domínios de problemas ((Das 2003), (Maurya *et al.* 2011), (Tatiraju & Mehta 2008) e (Guan *et al.* 2013)). Embora possa ser provado que o algoritmo K-Means sempre termina, ele não necessariamente encontra a configuração ótima de grupos além de ser também bastante sensível ao conjunto de centróides inicialmente escolhidos, escolha essa que, geralmente, é feita de maneira aleatória (Bottou & Bengio 1995).

O K-Means busca, iterativamente, diminuir a distância entre os dados e um conjunto específico de pontos i.e., os K centróides. O objetivo do algoritmo é encontrar a melhor partição dos dados de entrada em K grupos G_i (i = 1, 2,..., K), de maneira que a distância total entre os dados de um grupo e o seu respectivo centróide, somada com relação a todos os grupos, seja minimizada.

```
procedure K-Means

Entrada: CP = {E1, E2, ..., EN} {conjunto de pontos de dados a serem agrupados}
         K {número de grupos}
Saída: {G1,G2,...GK} {agrupamento constituído por K grupos de pontos de dados}

begin
  definir_centroides(CP,{C1, ..., CK})
  centroide_mudou ← true
  while centroide_mudou do
    begin
      for i = 1 to N do
        begin
         encontrar_centroide_mais_perto(Ei,{C1, ..., CK},Cmais_perto)
         Gmais_perto ← Ei
       end
     recalcular_centroides({C1, ..., CK},{G1,G2,...GK},{NC1, ..., NCK})
     checa_mudanca_centroide ({C1, ..., CK},{NC1, ..., NCK},centroide_mudou)
    end
end
return({C1, ..., CK})
end_procedure
```

Algoritmo 3.1 Descrição alto nível do K-Means.

Como descrito em (Jain *et al.* 1999), o algoritmo começa com o estabelecimento de uma partição inicial aleatória (função da escolha aleatória de um conjunto de K pontos que serão os centróides iniciais) e segue (a) associando cada um dos pontos do conjunto dados ao centróide que lhe for mais próximo, (b) recalculando os centróides (com base no conjunto de pontos que representa) e, então, voltando a realizar ambos procedimentos (a) e (b) até que uma condição de parada seja satisfeita.

Tal condição é, geralmente, definida quando todos os pontos exibem estabilidade, que é evidenciada por permanência nos respectivos grupos em que já estão (i.e., não acontecem mais transferências de pontos de um grupo para outro) ou, alternativamente, os centróides se mantêm inalterados. O K-Means é popular devido tanto à facilidade com que é implementado quanto à sua ordem de complexidade ser O(N), sendo N o número de dados que são entrada para o algoritmo.

A Figura 3.1 ilustra o funcionamento do K-Means sobre o conjunto de dados cujos elementos são representados por pontos (•) e os grupos por círculos que contornam partes do conjunto de pontos, considerando K=3. A Figura 3.1(a) exibe o conjunto de 13 pontos a serem agrupados (círculos), incluindo os três centróides escolhidos aleatoriamente (×). Na Figura 3.1(b) cada um dos pontos é associado ao centróide que lhe é mais próximo definindo, dessa forma, a primeira participação do conjunto inicial de pontos em grupos aleatórios. A Figura 3.1(c) exibe o resultado do procedimento de recálculo do centróide (como a média dos pontos que representa), provocando os deslocamentos e, então, o processo volta a ser repetido a partir da situação mostrada na Figura 3.1(d) até que pontos se estabilizem nos grupos aos quais pertencem.

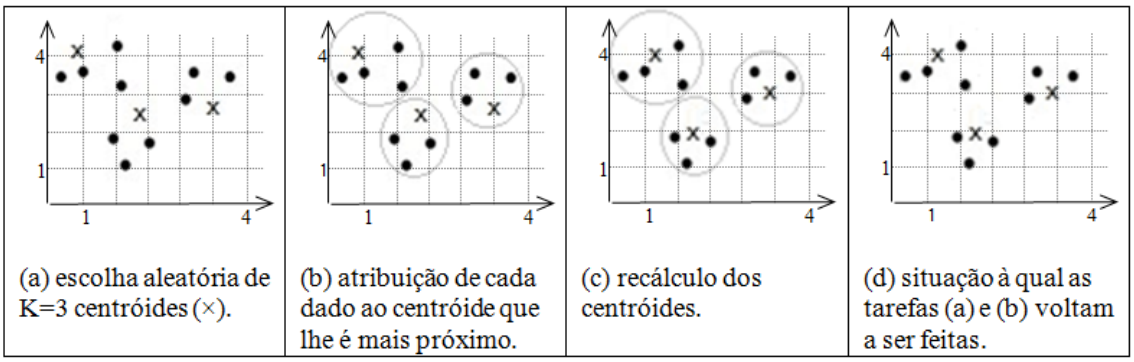


Figura 3.1 Resumo dos passos do algoritmo *K-Means*.

Apesar de ser considerado um algoritmo simples e eficiente, o K-Means possui algumas desvantagens. Uma delas é consequência da escolha aleatória inicial dos K centróides; devido à aleatoriedade do processo, podem ser escolhidos centróides muito próximos uns dos outros. Outra desvantagem é consequência do número fixo de grupos que deve ser informado ao algoritmo (como um valor de parâmetro).

Um valor pequeno associado ao parâmetro pode causar a junção de dois grupos que, em princípio, deveriam ser separados, enquanto que um número considerado grande pode fazer com que um grupo, que deveria ser único, seja particionado.

3.2 BSAS – *Basic Sequential Algorithmic Scheme*

O *Basic Sequential Algorithmic Scheme* (BSAS) foi apresentado em (Theodoridis & Koutroumbas 2009) como uma generalização da proposta descrita em (Hall 1967); sua descrição original em alto nível pode ser vista em Algoritmo 3.2 a seguir, na qual apenas a notação de instância de dado e de grupo foram alteradas para a notação adotada nessa dissertação.

Basic Sequential Algorithmic Scheme (BSAS)

- m = 1
- $G_m = \{E_1\}$
- **For** i = 2 **to** N
  - Encontrar G_k: $d(E_i,G_k) = \text{mim}_{1 \leq j \leq m}\, d(E_i,G_j)$
  - If ($d(E_i,G_k) > \Theta$) AND (m < q) then
    - * m = m+1
    - * $G_m = \{E_i\}$
  - Else
    - * $G_k = G_k \cup \{E_i\}$
    - * Quando necessário, atualizar representantes.
  - End (if}
- End {For}

Algoritmo 3.2 Pseudocódigo original do BSAS, como descrito em (Theodoridis & Koutroumbas 2009).

A descrição em Algoritmo 3.2 foi expandida e detalhada no pseudocódigo apresentado em Algoritmo 3.3. A entrada para o algoritmo BSAS é um conjunto de N dados $E = \{E_1, E_2, \ldots, E_N\}$ sendo cada E_i ($1 \leq i \leq N$) descrito por um vetor de M atributos. O algoritmo assume que valores associados a dois parâmetros sejam informados pelo usuário, a saber:

(1) um limite para o valor de dissimilaridade (Θ) e

(2) um limite para o número máximo de possíveis grupos a serem criados (q).

A cada iteração o algoritmo considera um próximo dado do conjunto de entrada E e, dependendo da distância do dado considerado aos grupos formados até o momento, executa uma das duas tarefas: (1) o incorpora a um dos grupos de dados já existentes ou (2) dá início à formação de um novo grupo, incluindo-o nele.

A ordem na qual os dados são apresentados ao BSAS tem influência direta no resultado final, tanto em relação ao número de grupos criados pelo algoritmo quanto em relação a quais dados cada um deles agrupa. O Algoritmo 3.2 detalha o procedimento BSAS que, dado um conjunto de dados e dois valores de parâmetros (q e Θ), agrupa os dados em (no máximo) q grupos, usando o valor fornecido (pelo usuário) do parâmetro de dissimilaridade Θ. No algoritmo (1) o conjunto de dados é apresentado ao algoritmo apenas uma vez e (2) o número de grupos não é conhecido *a priori*.

```
procedure BSAS
Entrada:
E: {E1, …, EN} {conjunto de N dados a serem agrupados}(1 ≤ i≤ N)
M: número de atributos que descrevem cada Ei (1 ≤ i≤ N) i.e., Ei = (Ei,1, Ei,2, ..., Ei,M)
Θ: limite de dissimilaridade
q : limite máximo para o número de grupos criados

Saída: G = {G1, G2, ..., Gz} {G: agrupamento de grupos de dados (1 ≤ z ≤ q)}

1. begin
2. G ← ∅
3. conta_grupo ← 1
4. Gconta_grupo ← {E1}            {criação do primeiro grupo participante do agrupamento G}
5. nconta_grupo ← 1
6. G ← G ∪ Gconta_grupo        (Agrupamento = conjunto de grupos; grupo=conjunto}
7. for i ← 2 to N do
8.   begin
9.     menor_distância ← distância(E1,G1)
10.     grupo_menor_distancia  ← 1
11.     for j ← 2 to conta_grupo do
12.       if d(Ei,Gj) < menor_distância then
13.         begin
14.           menor_distância ← distância(Ei,Gj)
15.           grupo_menor_distância  ← j
16.         end
17.     if (distância(Ei,Ggrupo_menor_distância) > Θ)
18.       then
19.         if (conta_grupo < q)                          {criação de novo grupo}
20.           then
21.             begin
22.               conta_grupo ← conta_grupo + 1
23.               Gconta_grupo ← {Ei}
24.               nconta_grupo ← 1
25.               G ← G ∪ Gconta_grupo
26.             end
27.           else send_message(' Reavaliar valor de q')        {caso tenha excedido q}
28.         else
29.           begin
30.             Ggrupo_menor_distância ← Ggrupo_menor_distância ∪ {Ei}
31.             n grupo_menor_distância ← n grupo_menor_distância + 1
32.           end
33.end
34. return(G)
```

Algoritmo 3.3 Pseudocódigo expandido do BSAS (*Basic Sequential Algorithmic Scheme*) que cria, no máximo, q grupos dedados, com base em um valor de dissimilaridade Θ.

Os valores dos dois parâmetros (q e Θ) têm também um papel relevante no agrupamento final obtido pelo BSAS. Se o valor atribuído a Θ for muito pequeno, grupos desnecessários podem ser criados e se for muito grande, um número reduzido

(aquém do número apropriado), será criado. Se o número máximo de grupos permitidos por agrupamento (q) não for estabelecido, o algoritmo irá criar tantos grupos quantos forem apropriados, de acordo com a lógica estabelecida em Algoritmo 3.3. Em uma situação, como a exemplificada na Figura 3.2(a), em que os dados perceptualmente se agrupam em três grupos, se o valor de q = 2, o BSAS será incapaz de evidenciar os três grupos, e nesse caso, provavelmente os dois grupos de pontos extremos à direita irão formar um único bloco, ilustrado na Figura 3.2(b). Para este mesmo exemplo se o valor de q não for estabelecido, provavelmente serão induzidos três grupos, com uma escolha conveniente do valor de Θ.

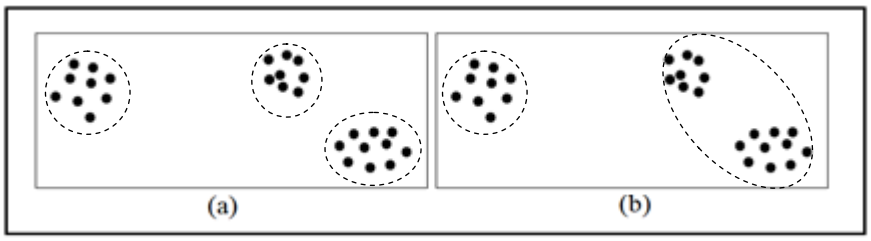


Figura 3.2 (a) Conjunto de dados no qual podem ser evidenciados (perceptualmente) três grupos. Na dependência dos valores dos parâmetros q e Θ, O BSAS pode induzir um agrupamento com um número diferente de 3; (b) Provável agrupamento se q = 2. Adaptada de (Theodoridis & Koutroumbas 2009).

Como discutido anteriormente, em um agrupamento um grupo pode ser representado por: (1) todas os dados que fazem parte dele (representação adotada por algoritmos baseados em *critérios locais* de agrupamentos) ou, então, (2) por um dado (que, via de regra, não é parte do conjunto de dados a ser agrupado) que representa todos os dados que estão no grupo, papel desempenhado pelo dado identificado como *representante (protótipo, representativo, centróide)* do grupo (representação adotada por algoritmos baseados em *critério global* de agrupamento).

Dependendo da aplicação a representação do grupo via dados que participam dele não é conveniente devido, principalmente, ao volume de dados a ser armazenado; nessa situação a representação via representante se torna mais conveniente. Dentre os esquemas de representação de grupos de dados discutidos no Capítulo 2 (Seção 2.2.1), esse trabalho vai adotar o mais popular deles i.e., aquele que representa um grupo de pontos pelo seu centróide (também referenciado na literatura como representante).

Como comentado em (Theodoridis & Koutroumbas 2009), (1) o BSAS pode ser usado com medida de similaridade ao invés de medida de dissimilaridade, desde que

modificado convenientemente, ou seja, o operador *min* deve ser substituído por *max*; (2) versões do BSAS que representam grupos por centróides, favorecem agrupamentos compactos e, portanto, não são recomendadas, caso exista evidência que outros tipos de agrupamentos estejam presentes e (3) o BSAS, como descrito em Algoritmo 3.1, pode ser relacionado à arquitetura neural criada pelo ART2 (*Adaptive Reasonance Theory*) (Carpenter & Grossberg 1987).

No Algoritmo 3.3 o procedimento *distância*(E_i,G_j) calcula a distância (ou dissimilaridade) entre um dado E_i (representado por um vetor de valores de atributos) a um grupo de dados G_j. Tal cálculo é função da representação adotada para grupo. Como visto no Capítulo 2, várias medidas podem ser empregadas nesse cálculo, levando a versões diferenciadas do algoritmo original (Algoritmo 3.2). Quando o grupo for representado por um centróide, o procedimento *distância*(E_i,G_j) vai estar determinando a distância entre dois dados: o primeiro argumento do procedimento (i.e., E_i) é efetivamente um dos dados do conjunto de dados a ser agrupado e o segundo argumento é um dado que representa o grupo G_j (R_{G_j}), ou seja, *distância*(E_i,R_{G_j}).

No caso em que um representante de um grupo é o centróide do grupo i.e., o dado definido como a média dos valores (de atributo) dos dados pertencentes ao grupo, a atualização do grupo (com a incorporação de uma nova instância E_i) consiste na modificação do centróide de acordo com a Equação (3.1).

$$R_{G_j}^{novo} = \frac{\left(n_{G_j}^{novo} - 1\right) R_{G_j}^{velho} + E_i}{n_{G_j}^{novo}} \qquad (3.1)$$

na qual: $n_{G_j}^{novo}$é a cardinalidade do grupo G_j após a incorporação do dado E_i, $R_{G_j}^{velho}$ e $R_{G_j}^{novo}$ são os centróides de G_j antes e após a incorporação de E_i ao grupo G_j.

3.3 O MBSAS – *Modified Basic Sequential Algorithmic Scheme*

Como pode ser verificado em Algoritmo 3.3, cada dado que é entrada para o algoritmo BSAS ou é incorporado a um grupo já criado ou, então, inicializa a formação de um novo grupo. A decisão de incorporar o dado a um grupo já existente ou, então, iniciar um novo grupo a partir dele, é tomada antes que o processo de estabelecimento de todos os grupos que compõem o agrupamento tenha finalizado (i.e., antes que todos

os dados tenham sido examinados). O *Modified Basic Sequential Algorithmic Scheme* (MBSAS) (Theodoridis & Koutroumbas 2009) pode ser considerado uma versão do BSAS na qual o processo de formação de grupos é refinado. Para tanto, em contrapartida, o conjunto de dados tem que ser processado duas vezes.

O pseudocódigo detalhado do MBSAS está descrito em Algoritmo 3.4 e consiste de duas fases. Na primeira fase alguns grupos são definidos com a incorporação de dados a eles; na segunda fase, os dados que não foram incorporados a qualquer dos grupos durante a primeira fase voltam a ser processados para identificar o grupo mais adequado ao qual possam ser agregados. A descrição original do MBSAS pode ser vista no Algoritmo 3.4 na qual apenas a notação de instância de dado e de grupo foram alteradas para a notação adotada nesse material; a descrição detalhada é apresentada em Algoritmo 3.5.

Modified Basic Sequential Algorithmic Scheme (MBSAS)

Fase I – Determinação do agrupamento

- $m = 1$
- $G_m = \{E_1\}$
- **For** i = 2 **to** N
  - Encontrar G_k: $d(E_i,G_k) = \text{mim}_{1 \leq j \leq m}\, d(E_i,G_j)$
  - If $(d(E_i,G_k) > \Theta)$ AND $(m < q)$ then
    - * $m = m+1$
    - * $G_m = \{E_i\}$
  - End (if}
- End {For}

Fase II – Classificação dos dados

- **For** i = 1 **to** N
  - If E_i não tenha sido atribuído a um grupo na *Fase I*, then
    - * Encontrar G_k: $d(E_i,G_k) = \text{mim}_{1 \leq j \leq m}\, d(E_i,G_j)$
    - * $G_k = G_k \cup \{E_i\}$
    - * Quando necessário, atualizar representantes.
  - End (if}
- End {For}

Algoritmo 3.4 Descrição original do MBSAS como apresentada em (Theodoridis & Koutroumbas 2009).

Em relação ao Algoritmo 3.4 é importante enfatizar que:

(1) o número de grupos é estabelecido e fixado na primeira fase do algoritmo (lembrando que tal número deve ser menor que o valor atribuído ao parâmetro q).

Na fase II, portanto, todos os grupos são levados em consideração quando dados que ainda não estão associadas a grupos são revisitados, na busca do grupo mais adequado para inseri-las.

(2) quando o algoritmo adota o conceito de centróide para representar cada um dos grupos do agrupamento, após a adição de um dado ao grupo, o centróide deve ser atualizado, seguindo Equação (3.1).

(3) similarmente ao BSAS, o MBSAS é também sensível à ordem na qual os dados são considerados.

(4) similarmente ao BSAS, o MBSAS pode adotar uma medida de similaridade, desde que uma pequena modificação seja feita.

```
procedure MBSAS
Entrada:
E: {E1, …, EN} {conjunto de N de dados a serem agrupados}(1 ≤ i≤ N)
M: número de atributos que descrevem cada Ei (1 ≤ i≤ N) i.e., Ei = (Ei,1, Ei,2, ..., Ei,M)
Θ: limite de dissimilaridade
q : limite máximo para o número de grupos criados
Saída: G = {G1, G2, ..., Gz} {G: agrupamento de grupos de dados (1 ≤ z ≤ q)}
1. begin
2. G ← ∅
3. conta_grupo ← 1
4. Gconta_grupo ← {E1}            {criação do primeiro grupo participante do agrupamento G}
5. nconta_grupo ← 1
6. G ← G ∪ Gconta_grupo          (Agrupamento = conjunto de grupos; grupo=conjunto}
7. for i ← 2 to N do
8.   begin
9.     menor_distância ← distância(E1,G1)
10.     grupo_menor_distancia  ← 1
11.     for j ← 2 to conta_grupo do
12.       if d(Ei,Gj) < menor_distância then
13.         begin
14.           menor_distância ← distância(Ei,Gj)
15.           grupo_menor_distância  ← j
16.         end
17.     if (distância(Ei,Grupo_menor_distância) > Θ)
18.       then
19.         if (conta_grupo < q)                          {criação de novo grupo}
20.           then
21.             begin
22.               conta_grupo ← conta_grupo + 1
23.               Gconta_grupo ← {Ei}
24.               nconta_grupo ← 1
25.               G ← G ∪ Gconta_grupo
26.             end
27.         end
28.  end
27. for i ← 1 to N do
28.   if (not pretence_a_algum_grupo(Ei))
29.     then
          menor_distância ← distância(E1,G1)
30.       grupo_menor_distancia  ← 1
31.       for j ← 2 to conta_grupo do
32.         if d(Ei,Gj) < menor_distância then
33.           begin
34.             menor_distância ← distância(Ei,Gj)
35.             grupo_menor_distância  ← j
36.           end
37.       end
38.       Gconta_grupo ← {Ei}
39.       nconta_grupo ← 1
40.     end
41. end
42.end
43. return(G).
```

Algoritmo 3.5 Descrição detalhada do MBSAS (*Modified Basic Sequential Algorithmic Scheme*). Sua primeira fase cria, no máximo, q grupos de dados, com base em um valor de dissimilaridade Θ. Na segunda fase os dados que não estão em qualquer dos grupos criados, voltam a ser processados com vistas a serem alocados a algum dos grupos criados.

3.4 O TTSAS – *Two-Threshold Sequential Algorithmic Scheme*

Como apontado anteriormente, tanto o BSAS quanto o MBSAS são dependentes tanto da ordem na qual os dados lhes são apresentados quanto do valor atribuído ao parâmetro Θ – valores não adequados de Θ podem implicar indução de agrupamentos não significativos. Uma maneira de contornar essas dificuldades é por meio da definição de uma região duvidosa (ver (Trahanias & Scordalakis 1989) e (Theodoridis & Koutroumbas 2009)), que pode ser implementada usando dois limites, Θ_1 e Θ_2 tal que $\Theta_2 > \Theta_1$ e pelas três regras descritas a seguir:

(1) Se o valor de dissimilaridade de um dado E_i ao grupo que lhe é mais próximo G_k for menor que o valor de Θ_1 (i.e., *distancia*$(E_i,G_k) < \Theta_1$), E_i é incorporado a G_k.

(2) Se o valor de dissimilaridade de um dado E_i ao grupo que lhe é mais próximo G_k for maior que o valor de Θ_2 (i.e., *distancia*$(E_i,G_k) > \Theta_2$), um novo grupo é inicializado com a inclusão de E_i nele.

(3) O fato da condição $\Theta_1 \leq$ *distancia*$(E_i,G_k) \leq \Theta_2$ ser verificada é indicativo que existe incerteza tanto na inclusão do dado em um grupo quanto na criação de um novo grupo e, portanto, a decisão de inclusão de E_i a um grupo (seja ele um novo grupo ou não) é postergada (ver pseudocódigo em Algoritmo 3.6 a seguir).

No pseudocódigo descrito no Algoritmo 3.6:

- conta_grupo: variável que representa o número de grupos criados até então
- função booleana *assigned*(E_i) definida por *assigned*: $E \rightarrow \{0,1\}$: $\forall\ E_i \in E = \{E_1, E_2, ..., E_N\}$, *assigned*$(E_i) = 0$ se E_i não fizer (ainda) parte de um grupo e *assigned*$(E_i) = 1$, se E_i já foi inserido em um grupo.
- não existe um limite para o número de grupos no agrupamento sendo induzido (i.e., q = N)
- exists_change verifica se existe pelo menos um dado que tenha sido atribuído a um grupo, em uma determinada iteração do comando **while**. A verificação é feita comparando o número de dados de E que foram atribuídos a grupos até a

finalização da iteração corrente (cur_change), com o número de dados que foram atribuídos a grupos até a iteração anterior (prev_change). Se exists_change = 0 significa que nenhum dado foi atribuído a um grupo durante a última iteração que 'varreu' E; o primeiro dado não atribuído é então usado para a formação de um novo grupo.

- O primeiro comando condicional que comparece no escopo do comando **for** garante que o algoritmo termina após o conjunto E ter sido 'varrido' no máximo N vezes (N execuções do comando **while**). Esta condição força o primeiro dado não atribuído ser inserido em um novo grupo, quando nenhum dado foi atribuído na última 'varrida' do conjunto E.
- Na prática o número de 'varridas' do conjunto E é muito menor que N. É importante mencionar que o esquema descrito pelo TTSAS é, quase sempre, pelo menos tão computacionalmente dispendioso quanto os dois anteriores (i.e., BSAS e MBSAS) porque, em geral, requer pelo menos duas 'varridas' do conjunto de dados E. Além disso, uma vez que a inserção de um dado a um grupo é adiada até que se consiga informação suficiente, o algoritmo acaba se tornando menos sensível à ordem de apresentação dos dados.
- De maneira semelhante aos dois algoritmos anteriores, também no TTSAS escolhas diferentes de medida de dissimilaridade entre um dado e um grupo levam a diferentes resultados. O TTSAS também favorece agrupamentos compactos, quando adota representantes para grupos.

É importante notar que nos três algoritmos, BSAS, MBSAS e TTSAS não ocorre situação de *deadlock*, ou seja, nenhum deles entra em um estado de execução em que existem dados que não foram atribuídos a grupos e que não podem ser atribuídos nem a grupos existentes ou tampouco a um novo grupo, independentemente do número de ´varridas´ dos dados. O BSAS garantidamente termina após uma única 'varrida' de E e o MBSAS após duas. No TTSAS a situação de *deadlock* é evitada uma vez que o algoritmo arbitrariamente atribui o primeiro dado não atribuído na iteração corrente a um novo grupo, caso nenhuma atribuição de dado tenha acontecido na iteração anterior.

```
procedure TTSAS
Entrada:
 E: {E1, …, EN} {conjunto de N de dados a serem agrupados}(1 ≤ i≤ N)
 M: número de atributos que descrevem cada Ei (1 ≤ i≤ N) i.e., Ei = (Ei,1, Ei,2, ..., Ei,M)
Θ1 e Θ2: limites inferior e superior de dissimilaridade, respectivamente

Saída: G = {G1, G2, ..., Gz} {G: agrupamento de grupos de dados (1 ≤ z ≤ N)}

1. begin
2.G ← ∅
3. for i ← 1 to N do assigned(Ei) ← 0
4.conta_grupo ← 0
5.prev_changes ← 0
6. cur_changes ← 0
7. exists_change ← 0
8. while ∃ assigned(Ei) = 0 (1 ≤ i ≤ N) do
9. begin
10.   for i ← 1 to N do
11.     if assigned(Ei) = 0 & first(Ei) & exists_change = 0
12.       then  begin
13.               conta_grupo ← conta_grupo + 1
14.               Gconta_grupo ← {Ei}
15.               G ← G ∪ Gconta_grupo
16.               assigned(Ei) ← 1
17.               cur_changes ← cur_changes + 1
18.             end
19.       else
20.         if assigned(Ei) = 0
21.             then begin
22.                   find Gk: d(Ei,Gk) = mim1 ≤j ≤ m distancia(Ei,Gj)
23.                    if d(Ei,Gk) < Θ1
24.                     then
25.                       begin
26.                         Gk = Gk ∪ {Ei}
27.                         assigned(Ei) ← 1
28.                         cur_changes ← cur_changes + 1
29.                       end
30.                    else
31.                      if d(Ei,Gk) > Θ2  then
32.                                        begin
33.                                          conta_grupo ← conta_grupo + 1
34.                                          Gconta_grupo = {Ei}
35.                                          G ← G ∪ Gconta_grupo
36.                                          assigned(Ei) ← 1
37.                                          cur_changes ← cur_changes + 1
38.                                        end
39.                    end
40.             else if assigned(Ei) = 1 then cur_changes ← cur_changes + 1
41. exists_change ← |cur_change – prev_change|
42.prev_change ← cur_change
43. cur_change ← 0
44.end {while}
45. return(G)
```

Algoritmo 3.6 Pseudocódigo expandido do TTSAS (*Two-Threshold Basic Sequential Algorithmic Scheme*), algoritmo que usa dois limites para valores de dissimilaridade (Θ_1 e Θ_2).

3.5 Estratégias de Refinamento Pós-Agrupamento

Embora tanto o MBSAS quanto o TTSAS serem considerados melhorias do BSAS, os resultados dos três algoritmos podem ainda ser melhorados: (1) quando o agrupamento resultante possui grupos que estão suficientemente próximos para serem unidos em único grupo e (2) com vistas a minimizar a sensibilidade à ordem dos dados (embora não tão crítico para o TTSAS).

Uma maneira de lidar com o problema (1) é por meio de um processo de pós-agrupamento, que une grupos considerados próximos o suficiente (de acordo com um parâmetro definido pelo usuário, *Close*), como proposto em (Fu *et al*. 1993) e descrito pelo procedimento *merge*, apresentado no Algoritmo 3.7.

A Figura 3.3 e a correspondente Tabela 3.1 mostram um exemplo no qual dois grupos são unidos, G_1 and G_4, uma vez que estão 'suficientemente próximos', ou seja, a distância entre seus centróides é menor que o valor do parâmtetro *Close*.

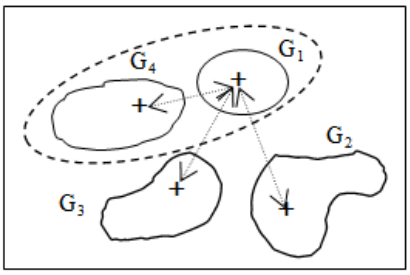


Figura 3.3 Agrupamento G={G_1, G_2, G_3, G_4} e a junção dos dois grupos mais próximos.

Tabela 3.1 Agrupamento G={G_1, G_2, G_3, G_4} e as distâncias (d_{ij}) entre os pares de centroides dos grupos.

G_i	G_j	d_{ij}	valor(d_{ij})
G_1	G_2	d_{12}	6
G_1	G_3	d_{13}	3
G_1	G_4	d_{14}	2
G_2	G_3	d_{23}	5
G_2	G_4	d_{24}	7
G_3	G_4	d_{34}	4

```
procedure merge
Entrada:
G: {G1, …, GZ} {saída do BSAS, MBSAS ou TTSAS}
Close: distância máxima permitida entre dois grupos de um
agrupamento, que, ainda, serão qualificados para junção.
Saída: G = {G1, G2, ..., GV} {resultado do agrupamento do
procedimento merge aplicado ao agrupamento original G =
{G1, G2, ..., GZ} (1 ≤ V ≤ Z)}

1. begin
2.  continue ← true
3.  while continue do
4.   begin
5.    indices_2groups_smaller_distance(G,i, j)
6.    if distance(Gi,Gj) < Close
7.      then begin
8.            Gi ← merge(Gi,Gj)
9.            G ← remove(G, Gj)
10.           RGi ← update(RGi)  {atualiza o centróide de Gi}
11.           for k ← j+1 to Z do
12.              begin
13.                rename(Gk,Gk-1)
14.                Z ← Z – 1
15.              end
16.          end
17.    else continue ← false
18. end
19. return(G)
```

Algoritmo 3.7 Pseudocódigo do procedimento *merge* que espera como entrada um agrupamento dado por G: $\{G_1, ..., G_Z\}$ e o parâmetro Close definido pelo usuário, que representa o quanto de proximidade dois grupos devem ter, para serem unidos.

O procedimento *merge* no Algoritmo 3.7 requer como entrada: (1) um agrupamento obtido por qualquer um dos algoritmos discutidos anteriormente, denotado como G: $\{G_1, ..., G_Z\}$ e (2) um valor definido pelo usuário para o parâmetro *Close*, que permite identificar em um agrupamento, os grupos que são próximos o suficiente para serem unidos em um só.

O procedimento *indices_2groups_smaller_distance()* identifica no agrupamento os dois grupos mais próximos dentre todos os pares de grupos do agrupamento. Se eles estão próximos o suficiente (de acordo com *Close*), eles são unidos em um único grupo e, em seguida, os grupos são renomeados. O processo é repetido até que não sejam mais detectados dois desses grupos. A Figura 3.4 mostra um exemplo simples de um agrupamento com quatro grupos (identificados por círculos e denominados ‘1’, ‘2’, ‘3’ e ‘4’). Após o procedimento *merge* medir as distâncias entre os quatro grupos (Figuras 3.4(a), 3.4(b) e 3.4(c)), a distância entre os grupos ‘1’ e ‘2’ foi a que estava de acordo com o parâmetro *Close* e assim foram unidos em um único grupo (Figura 3.4(d)). Os

grupos são renomeados e agora ficam identificados da seguinte forma (Figura 3.4(d)): grupos ‘1’ e ‘2’ como grupo ‘1’, grupo ‘3’ e grupo ‘4’ como grupo ‘2’ e grupo ‘3’, respectivamente.

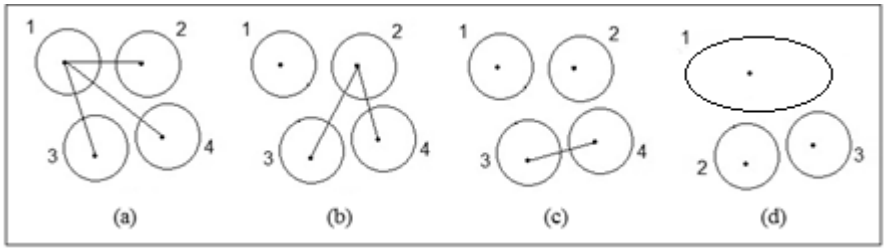


Figura 3.4 Renomeação dos grupos no procedimento *merge*.

Como sugerido em (Theodoridis & Koutroumbas 2009), uma forma de lidar com o problema (2) é por meio de um processo de pós-agrupamento (Algoritmo 3.8) que reatribui aqueles dados considerados deslocados – i.e., dados que, no agrupamento considerado, poderiam pertencer a outros grupos mais próximos àqueles aos quais pertencem.

A Figura 3.5 mostra a ideia básica do procedimento de reatribuição, onde é verificada para o dado E_i, alocado ao grupo ‘1’ (em um $G=\{G_1, G_2, G_3, G_4\}$), a menor distância entre ele e todos os centróides dos grupos do agrupamento (ilustrada pelas retas). Dependendo do grupo ao qual tal dado está atribuído e da distância verificada, é movido ou não a outro grupo. Assim, E_i é mantido ao grupo em que já está ou é reatribuído a outro e, nesse caso, o centróide do grupo deve ser atualizado.

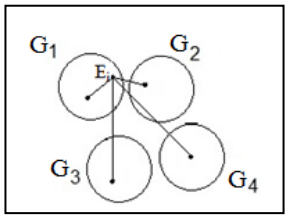


Figura 3.5 Agrupamento $G=\{G_1, G_2, G_3, G_4\}$ e o processo de reatribuição de um dado E_i.

O procedimento de refinamento (procedure *reassignment* – Algoritmo 3.8) requer como entrada: (1) um agrupamento obtido por qualquer um dos algoritmos discutidos anteriormente, denotado como $G=\{G_1, ..., G_Z\}$ e (2) o conjunto inicial dos dados. Para cada dado inicial E_i ($1 \leq i \leq N$) o procedimento *closest* identifica o centróide

do grupo mais próximo a ele e atribui o dado a este grupo (para a maioria dos dados a reatribuição não produzirá qualquer mudança). No próximo passo os centróides dos grupos são atualizados. Como um efeito colateral do procedimento *reassignment*, existe a possibilidade de um grupo terminando com nenhum dado. O último comando **for** no Algoritmo 3.8 o remove do agrupamento. Como esperado, o agrupamento resultante do procedimento *reassignment* pode ter um número menor de grupos do que o agrupamento de entrada.

```
procedure reassignment
Entrada:
CP: {E1, E2, ..., EN} {conjunto inicial de N dados}
G: {G1, …, GZ} {saída do BSAS, MBSAS ou TTSAS}
saída: G = {G1, G2, ..., GZ'} {resultado do procedimento
reassignment - Z' ≤ Z}

1. begin
2.   for i ← 1 to N do begin
3                        closest(Ei,G,j)
4.                        grupo(Ei) ← j
5.                       end
6.   for j ← 1 to Z do  begin
7.                         Gj ← {Ei ∈CP | grupo(Ei) = j}
8.                         update_representative(Gj)
9.                       end
10.  for j ← 1 to Z do if is_empty(Gj) then
11.                         begin
12.                           G ← G – {Gj}
13.                           Z ← Z – 1
14.                         end
15. end
16. return(G)
```

Algoritmo 3.8 Pseudocódigo do procedimento de reatribuição (procedure *reassignment*), que transfere dados deslocados a grupos mais próximos deles. Como entrada o algoritmo espera um agrupamento dado por G: $\{G_1, ..., G_Z\}$ e o conjunto inicial dos dados.

3.6 Avaliações das Propostas Evidenciadas na Literatura que Contemplam o Esquema Sequencial

Durante o levantamento e estudo de trabalhos relacionados a agrupamentos baseados no esquema sequencial, além daqueles já descritos neste capítulo, foram investigadas quatro propostas. Tais propostas, brevemente descritas a seguir, foram identificadas durante o levantamento bibliográfico para a elaboração do plano de pesquisa de mestrado e foram brevemente descritas no Exame de Qualificação. O que segue detalha uma avaliação de cada uma das quatro propostas; cada um dos quatro trabalhos foi lido e investigado, com os seguintes objetivos: (1) identificar e avaliar

possíveis contribuições e melhoramentos já contemplados na literatura e (2) analisar o desempenho de agrupamentos baseados no esquema sequencial em domínios de dados diferenciados.

(1) O trabalho descrito em (Ahmadi & Berangi 2008) utiliza o algoritmo TTSAS em uma aplicação específica, que é a da classificação de modulações (particularmente a modulação QAM (*Quadrature amplitude modulation*)), em um contexto de eletrônica e telecomunicação. O trabalho aborda o TTSAS em um sistema híbrido de aprendizado que também envolve a participação de uma rede neural. O entendimento mais detalhado do domínio da aplicação, bem como da maneira como a colaboração entre os dois métodos, agrupamento e neural, é implementada, vão permitir um melhor entendimento da versatilidade do algoritmo em esquemas colaborativos de aprendizado em um domínio de dados particularmente difícil (*stream data*).

As técnicas de reconhecimento de modulação são divididas em duas abordagens: o reconhecimento de sinais e o reconhecimento de seus símbolos. Um dos métodos típicos para o sinal modulado é a extração dos componentes *In-Phase* (I) e *Quad-Phase* (Q). Esses componentes podem ser vistos como um vetor no plano I/Q. O plano I/Q se refere como o diagrama de constelação da modulação. Com o uso deste diagrama, a classificação da modulação pode ser investigada como um problema de reconhecimento de padrões. A ideia é que, mediante a obtenção do número de grupos criados no plano I/Q, os níveis e o tipo de modulação podem ser identificados.

Neste trabalho, para agrupar os dados de símbolos no plano I/Q, foi utilizado o algoritmo TTSAS implementado com rede neural Hamming. Esta rede consite de duas camadas: (1) um gerador de pontuação e (2) usada para escolher a melhor pontuação. Os pesos na primeira camada representam os centróides dos grupos criados, que são atualizados a cada iteração da formação (do treinamento) da rede. Até o final desta formação os pesos ideais (i.e., grupos) podem ser alcançados. No início não há necessidade de conhecimento prévio do número de grupos, mas são necessários o número máximo de grupos e dois limiares ($\Theta_1 < \Theta_2$).

A descrição geral desse sistema híbrido (TTSAS e rede neural Hamming) pode ser resumida da seguinte forma. O algoritmo inicia com um único grupo e este grupo é considerado como um nó na rede neural Hamming. Em seguida o primeiro dado de

símbolo é aplicado à rede, definido como o centróide do primeiro grupo. Após isso, os demais símbolos são sequencialmente aplicados à rede. A classificação de um símbolo pode ocorrer: (1) a um dos grupos disponíveis, (2) é criado um novo grupo e o dado é classificado a ele ou (3) adiada. Em (1) e (2) a classe dos símbolos são definidas como um e em (3) como zero. O algoritmo é repetido até que todos os dados sejam classificados ou no caso de nenhum dado ser classificado numa iteração. O algoritmo utiliza a distância Euclidiana entre dado de símbolo e centróide, sendo escolhida a menor distância entre o dado de símbolo e um dos centróides. Se a distância $< \Theta_1$, o dado é atribuído ao grupo correspondente e o centróide é atualizado pela média, se a distância $> \Theta_2$ e o número máximo de grupos não tiver sido atingido, é criado um novo grupo e o dado é atribuído a ele como sendo o centróide do novo grupo e, se $\Theta_1 <$ distância $< \Theta_2$, a atribuição é adiada para próximas iterações.

Após o término do procedimento de agrupamento, é realizada uma etapa de refinamento realizada da seguinte forma: (1) os grupos com poucos elementos são eliminados e (2) a fusão de grupos considerados próximos o suficiente. Em (2), as distâncias de pares de centróides são calculadas em cada iteração e os dois grupos suficientemente mais próximos são unidos em um único grupo (o centróide é atualizado pela média). Isso é repetido até que o número de grupos seja igual ao número de pontos da constelação da modulação QAM (essa quantidade é conforme o tipo de modulação QAM).

O método proposto consiste no reconhecimento da modulação QAM usando TTSAS e a quantidade ideal de pontos da constelação. Inicialmente são determinados os pontos ideiais da constelação para cada tipo de modulação: 4 para 4-QAM, 16 para 16-QAM, 64 para 64-QAM e 256 para 256-QAM. Após isso os agrupamentos são determinados pelo algoritmo TTSAS. Durante a execução, os grupos resultantes são combinados com os pontos ideais dos modelos de modulações QAM. Os grupos resultantes são comparados com os pontos ideais da QAM (incialmente é caparada para 256-QAM pela distância Euclidiana). É calculada a distância Euclidiana entre os centróides dos grupos e os pontos ideais do tipo de modulação. Esta distância é comparada com Θ_1 (determinado inicialmente para 256-QAM), se a distância $< \Theta_1$, 256-QAM é reconhecida como a modulação e o algoritmo encerra, caso contrário, o próximo tipo (i.e., 64-QAM) é considerado para a avaliação e o algoritmo é executado

novamente. Este processo pode ir até 4-QAM, i.e, a avaliação começa com 256-QAM e termina com 4-QAM.

Os resultados dos experimentos do artigo mostram que o modelo obteve um desempenho eficiente e de alta precisão para o reconhecimento de modulação de vários tipos de QAM. Além disso, foi analisado que: a sensibilidade em relação ao ruído foi reduzida, o desempenho pode ser melhorado com o aumento do número de símbolos de dados e vantagens em relação aos cálculos finais dos centróides dos grupos e a determinação da localização destes centróides no diagrama de constelação.

(2) O trabalho descrito em (Mei & Lei 2008) investiga o uso do esquema sequencial de agrupamentos (particularmente a versão MBSAS) no domínio de processamento de imagens (mais especificamente, o de reconstrução do *background* em imagens). De maneira semelhante ao descrito em (1), espera-se que com um entendimento tanto do domínio de aplicação, quanto do uso do algoritmo, as contribuições do trabalho possam ser melhor entendidas e as vantagens do uso do algoritmo em tal domínio esclarecidas.

Dessa forma, o artigo propõe um esquema com três procedimentos para a reconstrução do *background* de imagens. O primeiro esquema é um algoritmo que tem como base o MBSAS (apresentado em (Theodoridis & Koutroumbas 2009)), denominado *Modified Basic Sequential Clustering* (MBSC). O MBSC é um refinamento do algoritmo BSC (*Basic Sequential Clustering*) proposto em (Xiao & Han 2007), semelhante ao BSAS (também apresentado em (Theodoridis & Koutroumbas 2009)). A ideia básica do BSC é a seguinte: inicia com a suposição de que o primeiro dado de entrada corresponde à classe inicial e o número da classe inicial é definido com "1". Cada dado considerado ou é atribuído a um grupo existente ou é atribuído a um grupo recentemente criado, dependendo das distâncias em relação aos grupos já criados. Portanto, as atribuições são definidas antes do agrupamento final ser estabelecido, no qual é determinado depois que todos os dados são apresentados. Portanto, o MBSC procura melhorar essa desvantagem (como o MBSAS o faz para o BSAS).

A reconstrução do *background* é uma abordagem para identificar objetos em movimento, especialmente para uma sequência em vídeo de uma câmera fixa. O

esquema inicial ocorre basicamente da seguinte maneira: (1) as intensidades de pixels em um período de tempo são classificadas pelo procedimento MBSC e (2) após (1) é executado um procedimento *Merging*, semelhante à estratégia de refinamento *merge* apresentada na Seção 3.5 deste capítulo. Os pixels cujas frequências de intensidade (aparição) são maiores do que o limiar, podem representar a cena como o modelo do *background*. Para reconstrução do *background* o esquema de agrupamento proposto conta os procedimentos MBSC, *merging* e *background selection* e segue quatro passos:

(1) Classificar as intensidades de pixels utilizando o procedimento MBSC. Neste algoritmo ocorrem duas fases: a primeira fase é a definição dos grupos, através da atribuição de alguns dados e, na segunda fase, os dados que não foram atribuídos durante a primeira fase, são apresentados pela segunda vez ao algoritmo e atribuídos aos grupos apropriados.

(2) Executar o procedimento *merging* após o término do MBSC, devido à possibilidade de acontecer de dois grupos criados estarem muito próximos e, nesse caso, podem ser juntados em um único grupo.

(3) Cálculo da frequência de intensidade de todos os grupos (assumindo os *p* grupos criados).

(4) Selecionar o pixel do *background* por meio do procedimento *background selection*, onde é escolhida uma única ou múltiplas imagens como as imagens do *background*. A ideia é separar os grupos de maior frequência de intensidade, calculada em (3), como sendo as imagens do *background*.

Como já discutido neste capítulo para os algoritmos sequenciais, aparentemente uma das possíveis dificuldades que podem ocorrer para o esquema utilizado em (Mei & Lei 2008), está na definição dos valores de parâmetros dos limiares. Visto que, para os três procedimentos (MBSC, *merging* e *background selection*), são necessários três limiares, Θ_1, Θ_2 e Θ_3, respectivamente. Os passos (1) e (2) ratificam as ideias já abordadas neste capítulo e realizadas durante os experimentos (i.e., algoritmos sequenciais e estratégias de refinamento). Os passos (3) e (4) são específicos para o domínio utilizado.

A conclusão desse trabalho é que para a reconstrução do *background* não é necessário o conhecimento prévio da cena, a atribuição para os dados é conseguida após

a criação dos grupos na primeira apresentação dos dados e as classes (grupos) mais próximas são evitadas por meio do procedimento *merging* (e o efeito da ordem da apresentação dos dados é reduzido).

(3) A proposta descrita em (Trahanias & Scordalakis 1989) busca contornar dificuldades relacionadas à forte dependência da ordem na qual dados são fornecidos ao BSAS e MBSAS. É apresentado um algoritmo sequencial de agrupamento semelhante à versão TTSAS apresentada em (Theodoridis & Koutroumbas 2009), com a proposta de ser mais eficiente e, tem como base, os algoritmos sequenciais de agrupamento convencionais. Inicialmente o artigo descreve algumas características desses algoritmos, tais como: (1) geralmente o primeiro dado é atribuído ao primeiro grupo e o *i*-ésimo dado é atribuído ao grupo mais próximo, a menos que a sua distância seja maior que o limiar definido (nesse caso um novo grupo é criado, respeitando o número máximo de grupos) e (2) pode conter pelo menos duas desvantagens, a dependência da ordem que os dados são apresentados e a sensibilidade ao limiar definido.

O trabalho propõe que as desvantagens (1) e (2) podem ser melhoradas (da mesma maneira como abordado para o algoritmo TTSAS na Seção 3.4 deste capítulo). A ideia do método, portanto, pode ser resumida da seguinte forma, no qual tem como entrada um dado E_i e dois limiares Θ_1 e Θ_2 ($\Theta_1 < \Theta_2$): (1) $E_i \in$ a um grupo G_j mais próximo se a distância $d(E_i,G_j) \leq \Theta_1$, (2) quando $d(E_i,G_j) \geq \Theta_2$, $E_i \notin G_j$ e (3) quando $\Theta_1 < d(E_i,G_j) < \Theta_2$, a atribuição de E_i pode ser adiada. O algoritmo utiliza para calcular $d(E_i,G_j)$ a distância Euclidiana.

No algoritmo apresentado é encontrado um procedimento *store* para armazenar o dado (E_i) que teve sua atribuíção adiada (caso (3), i.e., $\Theta_1 < d(E_i,G_j) < \Theta_2$). O trabalho evidencia que é conveniente considerar *store* como sendo uma estrutura de dados fila (respeitando a política *first-in*, *first-out* para os dados que forem colocados nessa estrutura) e, assim, isto mantém a ordem em que os dados foram inicialmente apresentados. Quando o número de dados a serem agrupados é grande, a possibilidade do número de dados a serem armazenados em *store* também é grande e, nesse caso, é necessário mais memória. Como alternativa, pode ser também usado um *buffer*, no qual os dados são reapresentados a cada vez que ele ficar cheio e com isso não é necessário

esperar até que todos os dados de entrada tenham sidos apresentados para reapresentá-los novamente.

A ideia de *store* como uma estrutura de dados fila é semelhante com o que foi abordado e implementado para o algoritmo TTSAS, i.e., nesse caso como os dados tem suas entradas organizadas em uma matriz que representa os vetores de pares *atributo-valor*, a 'varredura' sobre os dados também mantém a ordem em que os dados foram inicialmente apresentados, visto que o início dessa 'varredura' e a verificação se a atribuição foi adiada ocorre a partir do início da matriz, ou seja, do primeiro dado. Já a ideia de *store* como um *buffer* (limitado e menor que a quantidade N de dados) aparentemente pode resultar em uma melhoria na 'varredura' (sendo ela 'antecipada') e consequentemente na atribuição dos dados, entretando, para mensurar esta condição é necessária uma investigação empírica mais detalhada.

Contudo, o trabalho conclui que os problemas da ordem em que os dados são apresentados e a sensibilidade ao valor do limiar são eficientemente contornados pelo método proposto, como investifgado para o TTSAS. Além disso, sua implementação é simples e direta, podendo ser aplicado independente do esquema utilizado para representação dos dados e/ou cálculo de distância. Este método tem maior complexidade de tempo em relação ao método convencional de agrupamento sequencial, no entanto, o número de passos adicionais necessários, na pratica, é pequeno (poucos dados iniciais têm que ser reexaminados nesses passos). Portanto, a estimativa de valores adequados para dois limiares é mais fácil do para apenas um limiar (como ocorre para o BSAS e MBSAS). Além disso, a ordem de apresentação dos dados não é crucial devido as atribuições serem adiadas no caso de incerteza ($\Theta_1 < d(E_i,G_j) < \Theta_2$).

(4) O trabalho apresentado em (Liu *et al.* 2011) descreve o que os autores qualificam como um melhoramento da abordagem sequencial de agrupamentos, por meio da qual deficiências dos algoritmos existentes foram removidas usando o *bisecting k-means* (Jain *et al.* 1999). Especificamente é apresentado o algoritmo chamado *IGSClu* para agrupamento de sequências.

O trabalho utiliza definições e propriedades descritas em (Dai *et al*. 2010), que apresenta o *GSClu* (*Clustering Algorithms Based on Global*). A principal ideia abordada

no trabalho que apresenta o *GSClu* está em relação aos algoritmos de agrupamento de sequências que são baseados na hipótese de que a sequência pode ser caracterizada por suas características locais, sem diferenciar similaridade global e similaridade local, de diferentes aplicações (tais como o DNA e as proteínas com sub-padrões). No entanto, em alguns domínios de aplicação (tais como o utilizado neste trabalho de (Liu *et al.* 2011) (dados reais de vendas do comércio varejista)) e, devido às possíveis dificuldades na formação da frequência de sub-padrões, é mais razoável agrupar esses dados com base na similaridade global, como proposto para o *GSClu*.

O *GSClu* utiliza o *bisecting k-means* (que é basicamente uma extensão do *k-means),* que possibilita sua aplicação em conjuntos de dados maiores, inclusive em sequências. Os autores em (Jain *et al*. 1999) afirmam que algumas técnicas tentam selecionar uma boa partição inicial afim de tornar mais provável encontrar o valor mínimo global, e outra variação permite repartir e unir grupos resultantes. O processo do *bisecting k-means* é definido, resumidamente, seguindo quatro passos (Steinbach *et al*. 2000): (1) selecionar um grupo para dividir, (2) executa-se o *k-means* definindo apenas dois grupos, i.e., encontrar 2 'subgrupos' usando o algoritmo *k-means* básico; (3) escolhe-se o grupo com maior cardinalidade e então o divide em dois outros grupos, (a ideia é repetir o passo (2) por uma quantidade fixa de vezes e escolher a divisão que produzir o grupo com a maior similaridade global (para cada grupo, sua similaridade é a similaridade média de pares de sequências) e (4) repetir os passos 1, 2 e 3 até que o número desejado de clusters seja alcançado.

No *IGSCLu*, descrito neste trabalho de (Liu *et al.* 2011), também é utilizado o *bisecting k-means* que, segundo os autores, tem uma complexidade de tempo linear com o número de sequências. Este algoritmo tem como entrada um conjunto de dados SS com todas as sequências e o número k de grupos e, como saída, os k grupos de sequências. Dado que um vetor V_s representa as informações básicas sobre as letras e suas posições e que depois de digitalizar todas as sequências, há um vetor V_s para cada sequência, o *IGSClu* inicialmente faz isso para SS e cria um vetor V_s para cada sequência. Em seguida, é criado um único grupo com todas as sequências (inicialmente CN=1, onde CN é o número do grupo). O algoritmo mantém o invariante *while(CN<k)* e a cada iteração é realizado o processo do *bisecting k-means* que consiste em encontrar dois pontos centrais em um grupo P (na primeira iteração é um único grupo com todas

as sequências), que são CO1 e CO2, a maior e a menor sequência, assim é computada a distância entre eles e são criados dois novos grupos, e.g. P1 e P2.

Os experimentos ocorreram resumidamente da seguinte forma: dado um alfabeto $\sum$ com cardinalidade 10 ($\sum = (a_1, a_2, ..., a_T)$ e $T = |\sum|$ é a cardinalidade). Em primeiro lugar foram geradas k sequências S ($S = s_1\ s_2\ ...\ s_n$, $s_i \in \sum(1 \leq i \leq n)$, onde s_i é um elemento de S e $n = |S|$ é o tamanho da sequência S; $V_s = \{n_1, n_2, ..., n_T, d_1, d_2, ..., d_T\}$, onde $n_i(1 \leq i \leq T)$ e $d_i(1 \leq i \leq T)$ que é a distância da letra a_i a primeira letra de S) com diferentes tamanhos, que representam k grupos (chamadas de sequências sementes). Em seguida, foram geradas outras sequências através da alteração de tamanhos e de elementos nas sequências sementes. Os dados experimentados foram descritos como $K5C5000L100\Delta50VL5VP10$, onde K representa os K grupos, C é o número de sequências, L é a referência de tamanho, $\Delta50$ representa a base de alterações de comprimentos de cinco sequências sementes: 100, 150, 200, 250 e 300. Os resultados mostram, por meio de gráficos, um comparativo do *IGSClu* com o *GSClu* (em um dos resultados é visto que o *IGSClu* obteve um tempo de resposta menor para diferentes conjuntos (denominados ds1~ds6), que se diferenciam por apresentar sequências mais próximas umas das outras, e quanto maior essa proximidade menor foi o tempo de resposta comparado ao *GSClu*).

Este trabalho apresentou o *IGSCLU* como um melhoramento do *GSClu*. Neste melhoramento incluem os cálculos das distâncias e detecção de distâncias de subsequências, que podem reduzir a complexidade de tempo (apenas quando o tamanho das sequências estão próximas e as distâncias são menores do que as suas diferenças de tamanho). O *IGSClu* manteve o uso do *bisecting k-means* para o particionamento dos grupos, evidenciando seu bom desempenho quando utilizado em conjuntos de dados de sequências e motivando o interesse de um planejmanto do seu uso para outros estudos e/ou domínios. A conclusão do trabalho traz que o método pode ser apropriado para o agrupamento sequencial (de maneira mais rápida) e propõe como trabalho futuro utilizar este método na biologia para descobrir sequências *indels* (inserções e deleções de nucleotídios, i.e. *indels* (INserções e DELeções) são trechos de dois ou mais nucleotídeos que podem estar presentes (inserção) ou ausentes (deleção) nas fitas de DNA de um indivíduo).

Capítulo 4. Pré-processamento de Dados e Medidas de Validação

Este capítulo aborda dois aspectos subjacentes à área de aprendizado não supervisionado que foram considerados e estudados durante a pesquisa desenvolvida, com algumas de suas técnicas implementadas como subsistemas do sistema SEQ_CLUSTER: (1) o pré-processamento de dados, como um processo que antecede o uso de técnicas de aprendizado, com vistas a tratar os dados disponibilizados ao aprendizado e (2) o processo de validação de resultados obtidos, no contexto de algoritmos de agrupamento. Vale observar que em ambos os assuntos a pesquisa realizada foi motivada pelo interesse em abordar os algoritmos sequenciais de agrupamento em um contexto de experimentação com algoritmos de agrupamento. Assim sendo, o trabalho contempla também a implementação de dois métodos para tratamento de valores ausentes, uma situação corriqueira em pré-processamento de dados, bem como disponibiliza a implementação de três índices de validação, que auxiliam tanto na busca de um agrupamento conveniente quanto na determinação de valores adequados para determinados parâmetros de algoritmos sequenciais.

4.1 Pré-processamento de Dados

Via de regra dados coletados empiricamente advindos das mais variadas aplicações, e.g. exames clínicos, experimentos realizados em laboratórios, etc, que são utilizados por algoritmos de AM, devem passar por um tratamento, com o objetivo de, sempre que possível, eliminar erros tais como erro de leitura de equipamento, erro de imprecisão, erro de digitação, erro do próprio equipamento, erros de transmissão, alteração de valores devido à queda de energia, etc. Embora muitos desses erros sejam difíceis de serem detectados, muitos outros podem ser detectados com base na informação sobre os intervalos de valores associados a cada atributo que descreve os dados.

Dados reais, i.e., dados advindos de aplicações reais, em oposição àqueles artificialmente gerados, geralmente de maneira automática, com o intuito de criar um conjunto de dados satisfazendo determinadas propriedades, podem, também, ser fornecidos de maneira incompleta, como seria o caso de um dado descrito por valores de vários atributos relacionados a um exame de sangue, na qual o valor associado a

determina característica, não foi informado por alguma razão. Particularmente, grandes bases de dados que contém dados originários de múltiplas (e, geralmente, heterogêneas) fontes de dados reais, têm a tendência de armazenar dados com ruído, dados incompletos, dados inconsistentes, etc. Técnicas para tratar muitos desses problemas em dados são conhecidas como técnicas de pré-processamento de dados. Com o objetivo de organizá-las, os autores em (Han & Kamber 2006) propõem abordar tais técnicas em quatro categorias, usando como critério de agrupamento a funcionalidade da técnica:

(1) *Limpeza*: geralmente usadas para remover ruídos nos dados ou corrigir inconsistências;

(2) *Integração*: viabilizam a combinação e/ou junção de dados provenientes de múltiplas fontes, tal como acontece em *data warehouses*;

(3) *Transformação*: viabilizam uma transformação dos dados de maneira a promover determinada(s) característica(s). É o caso, por exemplo, de técnicas de normalização de dados com o objetivo de promover a precisão e eficiência de algoritmos de aprendizado, particularmente aqueles que fazem uso de medida de distância (como é o caso de muitos algoritmos de agrupamento);

(4) *Redução*: promovem uma condensação dos dados, geralmente com o objetivo de diminuir o seu volume (e facilitar armazenamento e recuperação). Técnicas que identificam redundâncias, por exemplo, promovem redução em dados.

A Figura 4.1, extraída de (Han & Kamber 2006) e traduzida, resume as quatro categorias de pré-processamento de dados descritas. É importante mencionar que as quatro categorias de técnicas não são disjuntas, uma vez que podem ser articuladas entre si. Uma limpeza de dados, por exemplo, pode envolver transformações para corrigir dados errados por meio, por exemplo, da transformação do campo data de todos os dados, para um padrão único. É fato que técnicas de pré-processamento de dados podem melhorar substancialmente a qualidade dos dados brutos e, consequentemente, melhorar a qualidade de resultados de métodos de aprendizado de máquina, particularmente de algoritmos de agrupamento.

Considerando que dados do mundo real podem conter muitos ruídos, estarem incompletos ou inconsistentes ou, ainda, terem que ser customizados às aplicações computacionais que os irão usar, o emprego de técnicas de pré-processamento de dados

antes do efetivo uso desses dados por sistemas pode contribuir para uma melhoria na qualidade dos próprios dados, bem como dos resultados produzidos pelos sistemas que os usam. Um registro de empregados em uma firma, por exemplo, em que o campo *ocupação* não está preenchido, é uma situação de valor de atributo ausente. Já em um registro do mesmo tipo, com o campo *salário* preenchido como –2.000,$# é considerado um ruído, devido à presença de caracteres espúrios. Uma situação de inconsistência de dados é aquela em que, por exemplo, o campo *idade* do registro está preenchido com o valor 29 e o campo *nascimento* com o valor 13/12/2000.

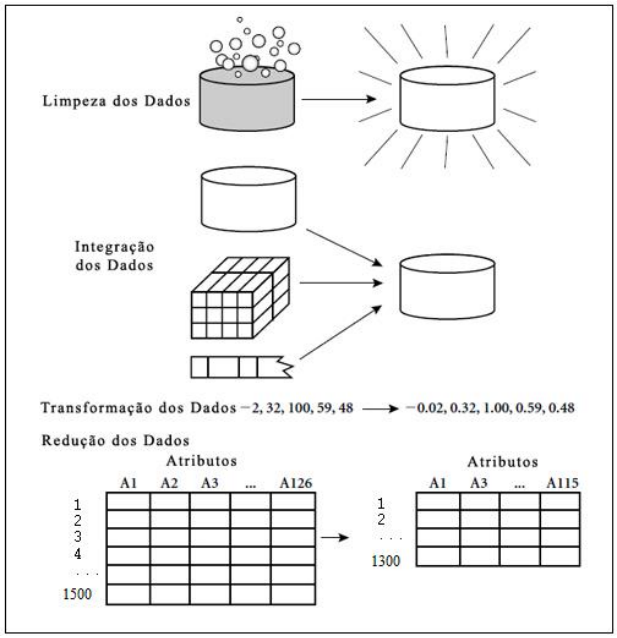


Figura 4.1 Representação pictórica das quatro categorias de técnicas de pré-processamento de dados, como propostas em (Han & Kamber 2006).

O pré-processamento de dados é, pois, uma etapa importante no processo de aprendizado indutivo de máquina; um algoritmo de aprendizado indutivo nada mais é do que um 'tradutor' de um conjunto de situações concretas (dados coletados/gerados) em uma expressão geral que os representa. A qualidade da expressão geral 'aprendida' é função da qualidade dos dados fornecidos ao algoritmo de aprendizado – dados com problemas muito provavelmente subsidiarão a indução de uma expressão geral também com problemas.

Considerando as quatro categorias de pré-processamento descritas (Figura 4.1) bem como as características do trabalho de pesquisa desenvolvido, essa seção tem

continuidade com foco em métodos enquadrados na categoria *Limpeza* e, dentre eles, métodos para tratar dados com valores ausentes. Como apontado em (Acuña & Rodriguez 2004) o problema do valor ausente é bastante comum em análise estatística. Volumes de dados em que 1% deles apresentam valores ausentes são tratados como problemas triviais, no intervalo 1–5% são considerados problemas administráveis, no intervalo 5–15% indicam necessidade de métodos sofisticados de tratamento de dados e superior a 15% representam um profundo impacto em qualquer possível interpretação a partir do volume de dados em questão. Muitas das bases de dados que disponibilizam conjuntos de dados geralmente assinalam a ausência de um valor de atributo por meio de um caractere especial, como é o caso do exemplo que segue. A Figura 4.2 mostra um extrato dos registros de dados de um dos arquivos de dados disponibilizados pelo UCI Repository-Irvine–UCLA (UCI Repository 2013), identificado como *Wisconsin Prognostic Breast Cancer* (*WPBC*). No extrato, que contém 10 dos 198 registros que compõem o WPBC, dois dos registros (em negrito) evidenciam uma situação de valor de atributo ausente (representado pelo símbolo ?).

843483,N,123,11.42,20.38,77.58,386.1,0.1425,0.2839,0.2414,0.1052,0.2597,0.09744,0.4956,1.156,3.445,27.23,0.00911,0.07458,0.05661,0.01867,0.05963,0.009208,14.91,26.5,98.87,567.7,0.2098,0.8663,0.6869,0.2575,0.6638,0.173,2,0

843584,R,27,20.29,14.34,135.1,1297,0.1003,0.1328,0.198,0.1043,0.1809,0.05883,0.7572,0.7813,5.438,94.44,0.01149,0.02461,0.05688,0.01885,0.01756,0.005115,22.54,16.67,152.2,1575,0.1374,0.205,0.4,0.1625,0.2364,0.07678,3.5,0

843786,R,77,12.75,15.29,84.6,502.7,0.1189,0.1569,0.1664,0.07666,0.1995,0.07164,0.3877,0.7402,2.999,30.85,0.007775,0.02987,0.04561,0.01357,0.01774,0.005114,15.51,20.37,107.3,733.2,0.1706,0.4196,0.5999,0.1709,0.3485,0.1179,2.5,0

844359,N,60,18.98,19.61,124.4,1112,0.09087,0.1237,0.1213,0.0891,0.1727,0.05767,0.5285,0.8434,3.592,61.21,0.003703,0.02354,0.02222,0.01332,0.01378,0.003926,23.39,25.45,152.6,1593,0.1144,0.3371,0.299,0.1922,0.2726,0.09581,1.5,?

844582,R,77,13.71,20.83,90.2,577.9,0.1189,0.1645,0.09366,0.05985,0.2196,0.07451,0.5835,1.377,3.856,50.96,0.008805,0.03029,0.02488,0.01448,0.01486,0.005412,17.06,28.14,110.6,897,0.1654,0.3682,0.2678,0.1556,0.3196,0.1151,4,10

844981,N,119,13,21.82,87.5,519.8,0.1273,0.1932,0.1859,0.09353,0.235,0.07389,0.3063,1.002,2.406,24.32,0.005731,0.03502,0.03553,0.01226,0.02143,0.003749,15.49,30.73,106.2,739.3,0.1703,0.5401,0.539,0.206,0.4378,0.1072,2,1

852973,N,52,15.3,25.27,102.4,732.4,0.1082,0.1697,0.1683,0.08751,0.1926,0.0654,0.439,1.012,3.498,43.5,0.005233,0.03057,0.03576,0.01083,0.01768,0.002967,20.27,36.71,149.3,1269,0.1641,0.611,0.6335,0.2024,0.4027,0.09876,2,0

854039,N,109,16.13,17.88,107,807.2,0.104,0.1559,0.1354,0.07752,0.1998,0.06515,0.334,0.6857,2.183,35.03,0.004185,0.02868,0.02664,0.009067,0.01703,0.003817,20.21,27.26,132.7,1261,0.1446,0.5804,0.5274,0.1864,0.427,0.1233,2.5,0

854253,N,12,16.74,21.59,110.1,869.5,0.0961,0.1336,0.1348,0.06018,0.1896,0.05656,0.4615,0.9197,3.008,45.19,0.005776,0.02499,0.03695,0.01195,0.02789,0.002665,20.01,29.02,133.5,1229,0.1563,0.3835,0.5409,0.1813,0.4863,0.08633,1.5,?

854268,N,31,14.25,21.72,93.63,633,0.09823,0.1098,0.1319,0.05598,0.1885,0.06125,0.286,1.019,2.657,24.91,0.005878,0.02995,0.04815,0.01161,0.02028,0.004022,15.89,30.36,116.2,799.6,0.1446,0.4238,0.5186,0.1447,0.3591,0.1014,3,13

Figura 4.2 Extrato do arquivo de dados *WPBC* no qual dois registros de dados têm valores ausentes (registros em negrito e valor ausente evidenciado por "?").

Cada registro de dado descreve um conjunto de informações relativas a um caso de câncer de mama. Informações são representadas como valores associados a um conjunto de atributos; valores foram determinados usando a imagem digitalizada do material coletado por uma punção na massa da mama – a maioria deles descreve características do núcleo das células presentes na imagem. O valor do último atributo que descreve os dados i.e., Situação dos nódulos linfáticos, está ausente em quatro dos 198 registros de dados. As principais informações sobre os atributos que descrevem os dados do *WPBC* estão na Tabela 4.1 e as informações completas podem ser consultadas na página do repositório, *http://archive.ics.uci.edu/ml/*

Tabela 4.1 Informações sobre o *WPBC* (#: número, RD: registros de dados, AT: atributos).

Características	Valores & Observações
# RD	198
# AT	35: identificação, situação, tempo, +32 atributos com valores reais.
AT_1: Número de identificação	Identificação do registro
AT_2: Situação	R (recorrência); N (não recorrência)
AT_3: Tempo	Se AT_2 = R, tempo da recorrência. Se AT_2 = N, tempo sem a doença.
AT_4 até AT_{33} (a) ráio (média das distâncias do centro a pontos no perímetro) (b) textura (desvio padrão de valores em escala-cinza) (c) perímetro (d) área (e) suavidade (f) compacidade (g) concavidade (severidade das porções côncavas do contorno) (h) pontos de concavidade (número de porções côncavas do contorno) (i) simetria (j) dimensão fractal	Dez atributos ((a) até (j)) com valores reais foram determinados a partir da análise de núcleos das células. Os 30 atributos que participam da descrição de cada registro de dado i.e., AT_4 até AT_{33} representam a média, desvio padrão e o pior (ou maior) (média dos três maiores valores) dos atributos (a) até (j) associados ao núcleo das células na imagem. Por exemplo, AT_4 representa a média dos ráios, AT_{14} o desvio padrão e AT_{24} a pior medida de ráio.
AT_{34}: Tamanho do Tumor	diâmetro do tumor extraído em centímetros
AT_{35} Situação dos nódulos linfáticos	número de nódulos linfáticos axilares positivos observados quando da cirurgia

Na literatura podem ser encontrados inúmeros trabalhos que abordam técnicas relacionadas ao problema de valor de atributo ausente, conhecidas como imputação. Em Estatística, imputação é definida como o processo conduzido para substituir ausência de valor associado a atributo(s), por um valor estatisticamente determinado. Métodos de

imputação têm sido propostos com base em inúmeros formalismos teóricos e constituem uma área de pesquisa que tem recebido inúmeras contribuições ao longo dos últimos anos. A maioria das técnicas de imputação podem ser categorizadas como *baseadas no modelo* ou como *não baseadas no modelo*.

Técnicas não baseadas em modelo agrupam métodos que fazem a substituição pela média/mediana ou, então, imputação *cold* ou *hot-deck*. Técnicas não baseadas no modelo são consistentes, fáceis de serem usadas, preservam os dados mas limitam sua variabilidade, aspecto importante que é contemplado por métodos baseados no modelo (Abdella & Marwala 2005). Técnicas baseadas no modelo incorporam algoritmos de aprendizado (redes neurais, por exemplo). Uma expressão geral do conceito é inferida com base no conjunto de dados (sem levar em consideração aqueles dados com valores ausentes) e, então, usando a expressão geral do conceito, os valores ausentes podem ser inferidos.

Considerando que valores ausentes de atributos constituem um problema quando da análise de dados, técnicas de imputação podem ser vistas como uma maneira de evitar os problemas ocasionados por um método popular de lidar com o problema, (muitas vezes considerado também entre as técnicas de imputação) conhecido como deleção de casos (*listwise deletion*) que tenham valores ausentes. A deleção de casos muitas vezes interfere no comportamento de técnicas estatísticas, uma vez que o poder de tais técnicas se deve, em parte, ao volume de dados amostrados. Como a deleção de casos exclui dados com valores ausentes, provoca uma redução da amostra sendo estatisticamente analisada. Como discutido em (Allison 2001), embora deleção de casos tenha alguns problemas tal método pode ainda ser preferível em relação a outros métodos disponíveis para tratamento de dados com valores ausentes.

A imputação *hot-deck* (HD) é um método estatístico que não é baseado em modelo e é um dos mais populares métodos de imputação de dados. Como apontado em (Blend & Marwala 2008) existem diferentes configurações de HD e todas elas, entretanto, baseadas na localização de dados similares àquele com valor ausente, no próprio conjunto de dados disponibilizado. O primeiro passo consiste em ordenar o conjunto de dados de acordo com um determinado número de variáveis, criando assim um conjunto ordenado. A técnica então localiza o primeiro dado que tenha valor ausente

de atributo e substitui o valor ausente pelo valor do mesmo atributo no dado anterior a ele. O processo é repetido para o(s) próximos valores ausentes encontrados, até que todos os valores ausentes sejam imputados. Já na imputação *cold-deck* (CD), os valores a serem imputados são extraídos de outro conjunto de dados. É fato, entretanto, que muitos métodos propostos de imputação introduzem um viés nos dados, ao imputarem valores que originalmente não estavam presentes.

Segue uma breve descrição de algumas técnicas de tratamento de valores ausentes abordadas nas referências (Han & Kamber 2006), (Acuña & Rodrigues 2004), (Blend & Marwala 2008) e (Zhang *et al*. 2012).

No livro (Han and Kamber 2006) são analisadas seis formas de tratar valores ausentes: (1) Ignorar o dado, (2) Preencher manualmente o valor ausente, (3) Usar uma constante global, (4) Usar a média dos valores presentes do atributo em questão, (5) Usar a média dos valores presentes do atributo em questão, considerando apenas dados que pertencem à mesma classe e (6) Usar o valor mais provável. Particularmente, o tratamento (2) é, na maioria das vezes, inviável. Na maioria dos casos a informação não está mais disponível ou é impraticável recuperá-la, ainda mais considerando o fator tempo (tal registro pode ser referente a uma situação de exame clínico ocorrido 2 anos atrás, por exemplo). O tratamento (3), que propõe o uso de uma constante global é também de pouca valia; o valor que está ausente seria substituído por um caractere com nenhum outro significado que o de representar um valor ausente.

Os autores em (Acuña & Rodrigues 2004) comentam que a presença de valores ausentes em um conjunto de dados pode afetar o desempenho de algoritmos de aprendizado que utilizam tais dados como conjunto de treinamento. Afirmam que vários métodos têm sido propostos para o tratamento de valores ausentes e o utilizado com maior frequência é o da deleção de casos (que contenham pelo menos um valor ausente). Na referência citada são apresentados quatro métodos diferentes para tratar valores ausentes, a saber: (1) Exclusão, que consiste em descartar aqueles dados (casos) que tenham pelo menos um valor ausente (i.e., deleção de casos), (2) Imputação média, em que o valor ausente de um atributo é substituído pela média dos valores do atributo em dados que compartilham a mesma classe, (3) Imputação Mediana, em que o valor ausente de um atributo é substituído pela mediana dos valores do atributo em dados que

compartilham a mesma classe e (4) Imputação KNN, em que o valor ausente de um atributo é substituído pelo valor médio do atributo dos vizinhos mais próximos do dado em que ele está ausente. Nesse caso, a mediana pode ser utilizada em vez da média.

Em (Blend & Marwala 2008) são apresentados e discutidos, comparativamente, os resultados de experimentação de técnicas de imputação baseadas em modelo implementadas por redes neurais auto-associativas, redes neuro fuzzy bem como de combinações híbridas desses métodos, com a imputação não baseada em modelo dada pela HD.

Como constatado em vários trabalhos de pesquisa, várias soluções têm sido sugeridas para resolver o problema de valores ausentes nos dados. A solução mais simples e a que ocorre com mais frequência é da eliminação de todos os dados que contenham pelo menos um valor ausente de atributo. Isso é possível quando há um conjunto de dados relativamente grande e os valores ausentes ocorrem com pouca frequência nos dados e foi uma das abordagens adotada nesse trabalho. A outra foi a da imputação do valor ausente pela média dos valores que comparecem nos outros dados do conjunto.

4.2 Medidas de Validação em Agrupamentos

Como enfatizado em muitas referências relacionadas a agrupamentos, um dos aspectos fundamentais da área está relacionado à avaliação dos resultados obtidos por um algoritmo de agrupamento, com vistas a encontrar a partição que melhor se adequa aos dados fornecidos. Para a avaliação da qualidade dos resultados obtidos usualmente são utilizados índices de validação. Vários estudos que investigam tais índices os organizam em três categorias distintas (Theodoridis and Koutroubas 2009) (Halkidi *et al*. 2001) (Halkidi *et al*. 2002a, 2002b) (Halkidi and Vazirgiannis 2001) (Kovacs *et al.* 2001): (1) critérios internos, (2) critérios externos e (3) critérios relativos.

Como comentado em (Kovacs *et al*. 2005), tanto os critérios internos quanto externos são baseados em métodos estatísticos e são, geralmente, computacionalmente custosos. Critérios externos avaliam o agrupamento com base em algum critério (geralmente intuitivo) fornecido pelo usuário enquanto que critérios internos fazem uso

de métricas aplicadas tanto ao conjunto de dados quanto ao método de agrupamento usado.

O critério relativo é direcionado pela comparação entre diferentes esquemas de agrupamento e usa uma das medidas externas ou internas. Um ou mais algoritmos de agrupamentos são executados múltiplas vezes com diferentes valores de parâmetros de entrada, mas tendo sempre como entrada o mesmo conjunto de dados. O objetivo do critério relativo é escolher o melhor esquema de agrupamento com base nos diferentes resultados obtidos.

A Tabela 4.2 identifica três índices de validação interna (e as publicações em que foram propostos), sendo que dois deles foram escolhidos para serem implementados e incorporados ao subsistema VALIDAÇÃO, parte integrante do sistema SEQ_CLUSTER. Este subsistema também disponibiliza de validação externa, que requer a informação da classe associada a cada dado.

Tabela 4.2 Índices de validação investigados e disponibilizados no SEQ_CLUSTER.

Índices	Referências	Seção
Dunn (D)	(Dunn 1974)	4.2.1
Davies-Bouldin (DB)	(Davies & Bouldin 1979)	4.2.2
Estatística Γ Modificada por Hubert	(Hubert & Arabie 1985)	4.2.3

Como o objetivo de tornar o capítulo auto-contido, a notação estabelecida volta a ser reapresentada na Tabela 4.3, para facilitar o entendimento das três próximas subseções, cada uma delas focalizando um índice de validação.

Tabela 4.3 Nomenclatura e notação utilizadas.

Nomenclatura utilizada	Notação
Conjunto dos pontos (de dados) a serem agrupados	CP
Número de elementos em CP (|CP|)	N
Centro do conjunto CP	centro
Número de atributos que descrevem cada um dos pontos	M
Número de grupos em um agrupamento	NG
I-ésimo grupo de um agrupamento	G_i
Número de pontos em um grupo G_i	n_i
Centro do grupo G_i	c_i
Vetor de variância de G_i	$\sigma(G_i)$
Pontos de dados	E_x, E_y
Distância entre dois pontos de dados	$dist(E_x, E_y)$
Distância entre dois grupos	$d(G_i,G_j)$
Norma de um ponto de dado E_x ($\|\|E_x\|\|$)	$(E_x^T . E_x)^{1/2}$

4.2.1 Índice de Dunn (D)

O índice de validação de Dunn (D), proposto em (Dunn 1974) busca identificar em um agrupamento *grupos compactos* e *bem separados*. Para isso, este índice utiliza a distância mínima entre dois dados que pertencem a diferentes grupos do agrupamento para caracterizar a separação inter-grupos e o diâmetro máximo entre todos os grupos do agrupamento como sendo a compactação intra-grupo. Uma maneira de equacionar esse índice é por meio da expressão mostrada na Equação (4.1).

$$\min_i \left\{ \min_j \left(\frac{\min_{E_x \in G_i,\, E_y \in G_j} \mathrm{dist}(E_x, E_y)}{\max_k \left\{ \max_{E_x,\, E_y \in G_k} \mathrm{dist}(E_x, E_y) \right\}} \right) \right\} \tag{4.1}$$

Outra maneira de representar o Índice de Dunn é por meio da definição formal dos conceitos de *distância entre grupos* e de *diâmetro de um grupo*, como estabelecem a Equação (4.2) e a Equação (4.3), respectivamente. Considere dois grupos de pontos G_i e G_j. A distância (d) entre esses grupos é dada por:

$$d(G_i, G_j) = \min_{E_x \in G_i, E_y \in G_j} \mathrm{dist}(E_x, E_y) \tag{4.2}$$

O diâmetro (*diam)* de um grupo G_i é definido pela Equação (4.3), que caracteriza o diâmetro de um grupo como a distância entre seus pontos mais distantes. O valor do diâmetro de um conjunto de pontos pode ser interpretado como uma medida da dispersão desse conjunto de pontos.

$$\mathrm{diam}(G_i) = \max_{E_x, E_y \in G_i} \mathrm{dist}(E_x, E_y) \tag{4.3}$$

O índice de Dunn para um agrupamento constituído de *M* grupos pode então ser reescrito como na Equação (4.4)

$$D_M = \min_{i=1,\ldots,M} \left\{ \min_{j=i+1,\ldots,M} \left(\frac{d(G_i, G_j)}{\max_{k=1,\ldots,M} \mathrm{diam}(G_k)} \right) \right\} \tag{4.4}$$

O valor de D_M pode ser interpretado da seguinte maneira: se o agrupamento gerado a partir de um conjunto de dados contém grupos compactos e bem separados, a

distância entre os grupos deve ser relativamente grande e o diâmetro dos grupos deve ser relativamente pequeno. Assim, com base na Equação (4.4), pode-se concluir que valores de D_M mais altos são indicativos da presença, no agrupamento, de grupos com tais características. Segundo (Halkidi *et al.* 2001), o índice D_M não apresenta qualquer tendência com relação ao número de grupos e sobre ele valem os seguintes comentários: (1) alto tempo envolvido em seu cálculo e (2) alta sensibilidade à presença de ruído nos dados, uma vez que ruídos, geralmente, tendem a aumentar o tamanho do diâmetro de um grupo de pontos (denominador da Equação (4.4)). Três outros índices baseados no índice de Dunn, foram propostos em (Pal & Biswas 1997) e são mais robustos com relação à presença de ruído nos dados. Para a especificação e uso de tais índices, entretanto, são necessários os conceitos de árvore *spanning minimal*, grafo de vizinhança relativa e grafo de Gabriel, que fogem ao escopo da investigação conduzida durante esse trabalho de mestrado. Como para esses três conceitos serem abordados com o cuidado que merecem implicaria um aprofundamento detalhado e, como estão relacionado à uma área subjacente à pesquisa conduzida, decidiu-se por investigá-los como uma continuidade ao trabalho desenvolvido.

4.2.2 Índice Davies-Bouldin (DB)

O índice Davies-Bouldin (DB) proposto em (Davies & Bouldin 1979) é baseado em uma medida de similaridade (R_{ij}) entre grupos (G_i e G_j, $i \neq j$ e i, j = 1, ..., NG) do agrupamento. Tal similaridade, por sua vez, se baseia na medida de dispersão de um grupo (s_i) (Equação 4.5) e na medida de dissimilaridade entre grupos (dis_{ij}) (Equação 4.6). A medida de similaridade R_{ij} entre grupos é dada pela Equação 4.7 e o índice DB pela Equação 4.8.

$$s_i = \frac{1}{n_i} \sum_{E \in G_i} \mathrm{dist}(E, c_i) \tag{4.5}$$

$$\mathrm{dis}_{ij} = d(c_i, c_j) \tag{4.6}$$

$$R_{ij} = \frac{s_i + s_j}{\mathrm{dis}_{ij}} \tag{4.7}$$

$$DB = \frac{1}{NG}\sum_{i=1}^{NG} R_i \text{, onde } R_i = \max_{j=1,\dots,NB; i\neq j}(R_{ij}), i = 1, \dots, NB \tag{4.8}$$

O DB mede a similaridade média entre cada grupo e aquele que lhe é mais semelhante. Como grupos supostamente devem ser compactos e separados, quanto mais baixo for o valor de DB, melhor é a configuração de grupos obtida. Como comentado em (Halkidi *et al*. 2001a), é desejável que os grupos tenham um mínimo de similaridade possível uns com os outros e, portanto, o que são buscados são agrupamentos que minimizam o valor de DB.

A medida de similaridade entre grupos (R_{ij}) pode ser definida de várias maneiras mas deve satisfazer as cinco condições listadas a seguir, como estabelecido em (Davies & Bouldin 1979):

(1) $R_{ij} \geq 0$

(2) $R_{ij} = R_{ji}$

(3) se $s_i = 0$ e $s_j = 0$ então $R_{ij} = 0$

(4) se $s_j > s_k$ e $d_{ij} = d_{ik}$ então $R_{ij} > R_{ik}$

(5) se $s_j = s_k$ e $d_{ij} < d_{ik}$ então $R_{ij} > R_{ik}$

4.2.3 Índice Estatística Γ Modificada por Hubert

O índice Γ modificado por Hubert (Γ), proposto em (Hubert & Arabie 1985) e descrito pela Equação (4.9) avalia a diferença entre grupos de um agrupamento por meio da contagem dos pares de dados que pertencem a diferentes grupos no agrupamento em questão.

$$\frac{2}{N(N-1)}\sum_{E_x \in CP}\sum_{E_y \in CP} \text{dist}(E_x, E_y)\,\text{dist}_{E_x \in G_i, E_y \in G_j}(c_i, c_j) \tag{4.9}$$

Note que o índice contabiliza, para cada par de pontos (E_x, E_y) do conjunto de pontos CP, o produto da distância entre eles pela distância entre os centros dos grupos aos quais pertencem. Para pontos E_x e E_y pertencentes ao mesmo grupo, os centros c_i e c_j são o mesmo e, portanto, a distância entre os centros será zero. Contribuem pois para o Γ apenas pares de pontos que pertencem a grupos distintos.

4.3 Considerações Finais

Este capítulo buscou contextualizar e evidenciar a relevância de duas áreas de pesquisa cujos resultados contribuem diretamente para o uso de técnicas de aprendizado não supervisionado, a saber: (1) pré-processamento de dados e (2) índices de validação. O capítulo aborda tais assuntos com suficientes detalhe apenas para proporcionar uma visão geral da importância de ambos em sistemas que se propõem a disponibilizar implementações de métodos de agrupamentos e detalhar um conjunto pequeno de técnicas que os viabilizam.

Tanto o pré-processamento de dados, como uma fase anterior ao processo de agrupamento, quanto a avaliação, como uma fase posterior ao processo de agrupamento contribuem para complementar uma tarefa de agrupamento de dados. O módulo PRÉ-PROCESSAMENTO e o módulo VALIDAÇÃO que implementam, respectivamente, o tratamento de valor ausente e a avaliação de resultado de algoritmo de agrupamento estão integrados ao ambiente do sistema SEQ_CLUSTER, como detalha o Capítulo 5.

Capítulo 5. O Sistema Computacional SEQ_CLUSTER

Este capítulo descreve o sistema computacional desenvolvido chamado SEQ_CLUSTER (*SEQuential algorithm based CLUSTERing*) que disponibiliza a implementação de três algoritmos de uma família de algoritmos de agrupamento (aqueles caracterizados como sequencias), dois métodos de refinamento e a combinação de ambos (como um processo de pós-agrupamento) e três métodos de validação de agrupamentos. A motivação para o desenvolvimento do SEQ_CLUSTER foi a de oferecer um ambiente computacional para estudo, investigação e experimentação de algoritmos de agrupamentos, particularmente (mas não exclusivamente) os sequenciais, como apresentados e discutidos no Capítulo 3. O sistema também disponibiliza uma implementação do algoritmo K-Means, para fins de comparação.

5.1 Características Básicas, Operacionalidade e Funcionalidades do SEQ_CLUSTER

O SEQ_CLUSTER é um sistema computacional desenvolvido em Delphi (*Borland Delphi Enterprise* versão 7 – build 4.453), executado sobre a plataforma Microsoft Windows. A arquitetura funcional do SEQ_CLUSTER está organizada em cinco módulos a saber: (1) *formatação de dados*, (2) *pré-processamento*; (3) *gerador de conjunto de dados*; (4) *agrupamento e (5) validação.*

(1) *formatação de dados*: auxilia no ajuste dos dados de entrada para um formato padrão único utilizado pelo módulo de agrupamento, quando necessário.

(2) *pré-processamento*: faz a remoção dos dados ou a substituição pela média dos atributos correspondentes de dados com problemas, tais como valores de atributos ausentes.

(3) *gerador de conjunto de dados*: responsável por criar conjuntos de dados sintéticos, direcionado pelo usuário, com base nas especificações do usuário.

(4) *agrupamento*: contém as implementações dos algoritmos de agrupamento BSAS, MBSAS, TTSAS e K_Means, e dos procedimentos de refinamento: *merge*, *reassignment* e *merge+reassignment*.

(5) *validação*: disponibiliza recursos para a realização de validação externa e dos seguintes índices de validação interna: de Dunn e de Davies-Bouldin.

Os módulos (2), (4) e (5) são os módulos principais que contemplam a motivação para o desenvolvimento do sistema proposto neste trabalho. Entretanto, é importante evidenciar que, como ocorre, por exemplo, para o *software Weka* desenvolvido pelo Machine Learning Project (Department of CS of The University of Waikato) e descrito em (Bouckaert *et al*. 2012), o SEQ_CLUSTER pressupõe que os dados de entrada estejam em um formato padrão único, o qual será apresentado mais adiante nesta seção. Com o objetivo de facilitar essa formatação (obviamente não para todos os casos), foi desenvolvida a primeira versão de uma ferramenta denominada '*formatação de dados*' que auxilia no ajuste dos dados de entrada, visto que em muitos conjuntos de dados, como, por exemplo, do UCI Repository, podem conter, além de outras situações adversas ao formato padrão, atributos de dados e seus valores separados por meio de espaços simples, tabulações, vírgulas, pontos, etc. Presentemente os módulos (1) e (3) ainda funcionam *stand-alone* e estão sendo atualizados e readaptados para integrar o SEQ_CLUSTER.

A Figura 5.1 mostra um fluxograma com as possíveis alternativas (e sequenciamento) de uso dos cinco módulos do SEQ_CLUSTER. A entrada do sistema é um conjunto de dados (CD) que pode ser pertencente ao repositório UCI ou sintético gerado artificialmente pelo módulo *gerador de dados*. O sistema está previsto para ser executado de três maneiras: (1) por meio da execução do módulo *formatação de dados*, caso a formatação de dados para o formato padrão de arquivo seja necessária (exceto para os arquivos de dados sintéticos que já são gerados no formato correto); (2) via pré-processamento, caso os dados utilizados tenham valores ausentes de atributo ou, então, (3) usando algum algoritmo de agrupamento em dados que se adequam ao padrão utilizado pelo sistema, bem como utilizar ou não alguma estratégia de refinamento pós-agrupamento e algum método de validação de agrupamento obtido.

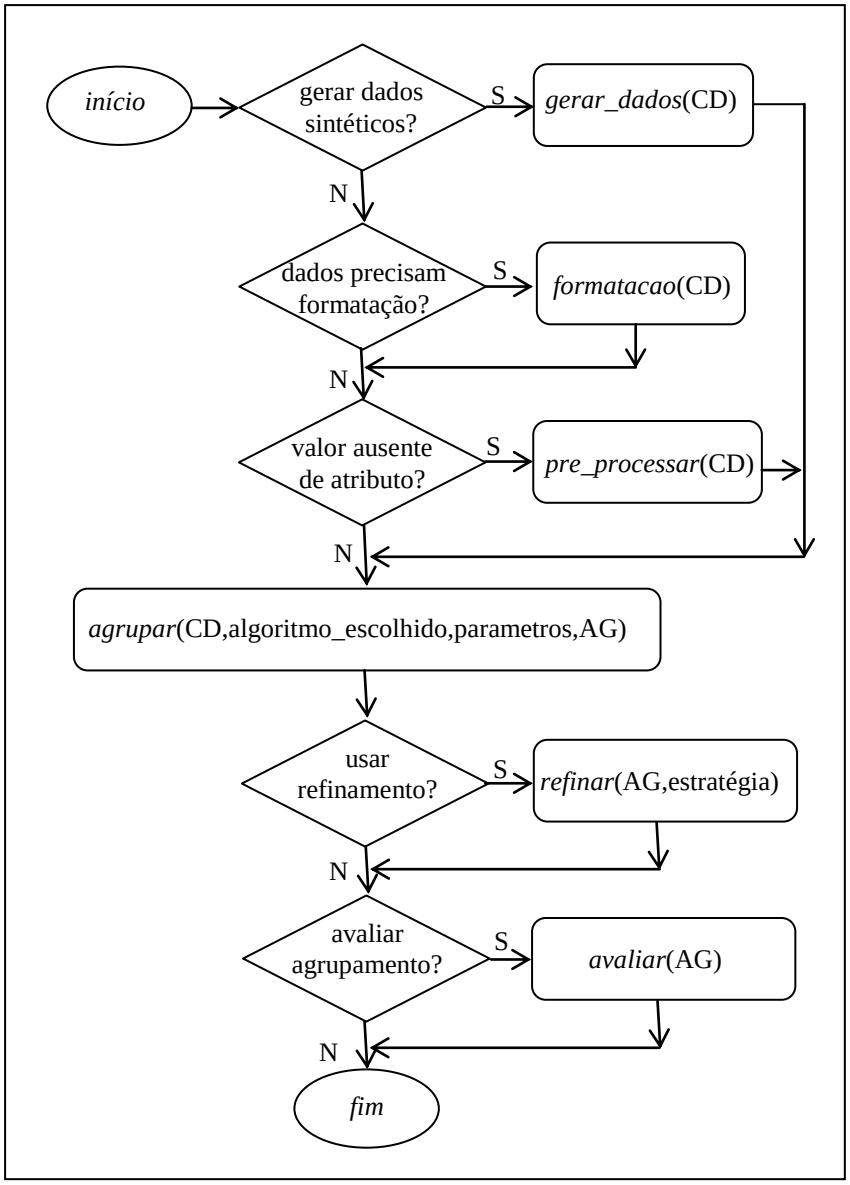


Figura 5.1 Ilustração para uso dos módulos do SEQ_CLUSTER.

O sistema disponibiliza ao usuário a utilização de uma sequência de cinco etapas para gerar um agrupamento, representadas pelas abas *Detalhamento*, *Arquivo de Dados*, *Algoritmo de Agrupamento e avaliação de resultados* e *Gráfico e Agrupamento dos Dados*, respectivamente.

A tela inicial do sistema, mostrada na Figura 5.2, é a aba *Detalhamento* (aba 1) que permite ao usuário escolher o nível de detalhamento na visualização dos dados contidos no arquivo de dados, a ser selecionado na aba *Arquivo de Dados* (aba 2). Os três possíveis níveis de detalhamento oferecidos ao usuário são:

(1) *Nível 1*: mostra o conteúdo completo original do arquivo contendo os dados, sem destacar separadamente os dados e seus atributos.

(2) *Nível 2:* similar ao *Nível 1* destacando as linhas de atributos;

(3) *Nível 3:* acrescenta, às visualizações anteriores, junto aos seus atributos, visualmente alocados em uma matriz que representa vetores de pares *atributo-valor*. A Figura 5.3 mostra as possibilidades de visualização do conjunto de dados de acordo com os três níveis.

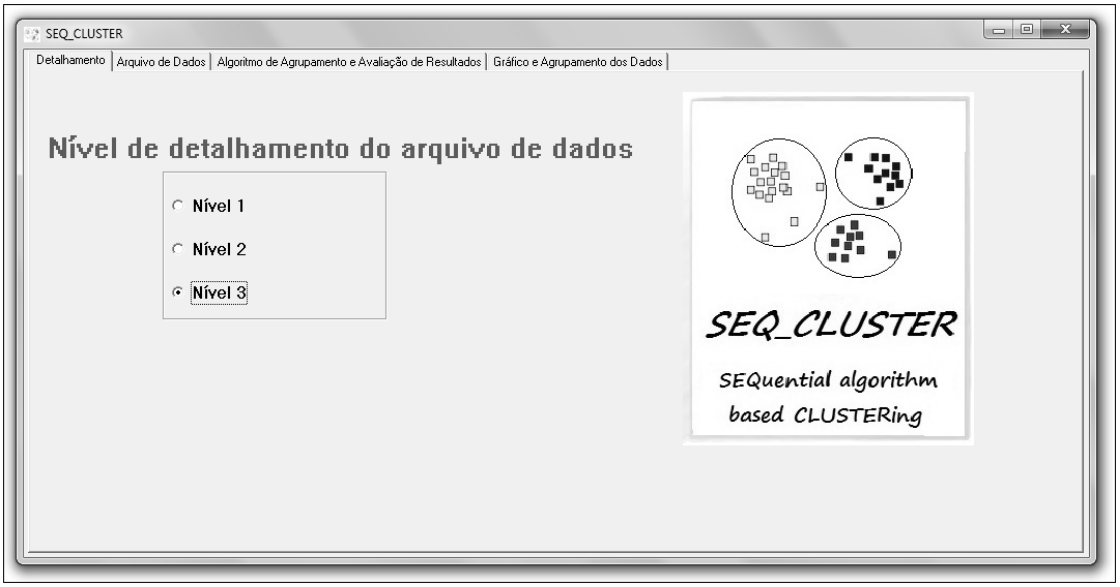


Figura 5.2 Tela inicial do sistema SEQ_CLUSTER.

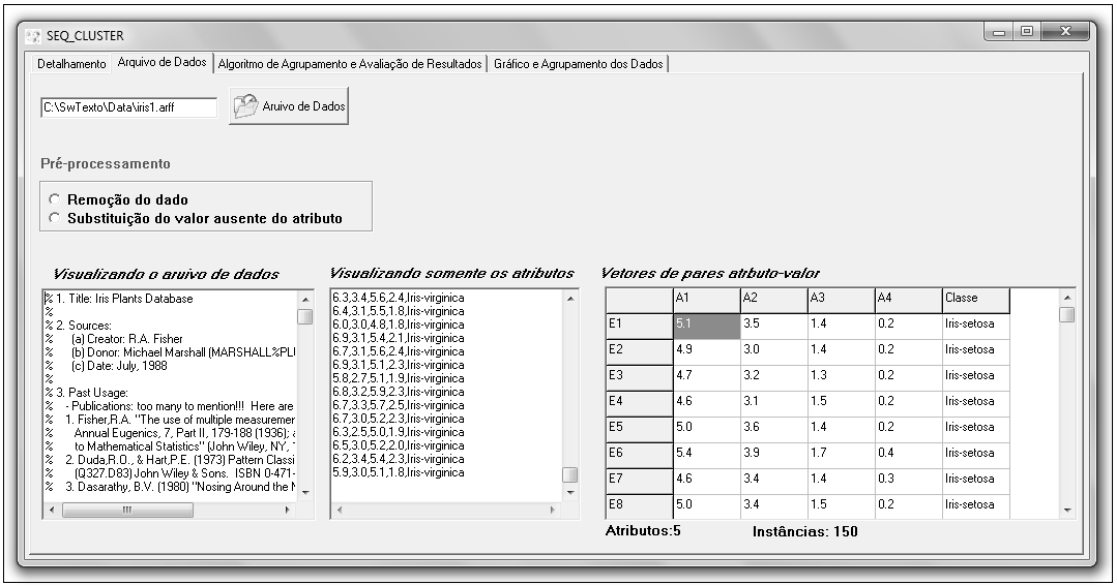


Figura 5.3 Os três níveis de visualização do conjunto de dados disponibilizados pelo SEQ_CLUSTER.

A Figura 5.3 mostra a aba *Arquivo de Dados* (aba 2), na qual estão disponíveis os dois possíveis métodos de pré-processamento dos dados oferecidos ao usuário, são eles:

(1) remoção do dado: consiste em descartar dados que tenham, pelo menos, um de seus atributos com valor ausente;

(2) substituição do valor ausente de atributo: imputação do valor ausente pela média dos valores do atributo correspondente que comparecem nos outros dados do conjunto.

Como pode ser visto ainda na Figura 5.3, a aba *Arquivo de Dados* permite a seleção do arquivo texto que contém o conjunto de dados, por meio do botão *Arquivo de Dados*. Para realizar essa seleção, é necessário que um arquivo texto em formato *.ARFF* ou .TXT esteja disponível no diretório *Datasets*. Um arquivo texto no formato *.ARFF* (*Attribute-Relation File Format*) é um arquivo em ASCII que descreve uma lista de dados formados pelo mesmo conjunto de atributos. Esses arquivos em formato *.ARFF* foram desenvolvidos como parte do Machine Learning Project (Department of CS of The University of Waikato), para serem usados pelos softwares desenvolvidos durante o projeto (ver http://www.cs.waikato.ac.nz/ml/weka/arff.html para detalhes). Da mesma forma, um arquivo texto no formato .TXT é um arquivo em ASCII que descreve uma lista de dados formados pelo mesmo conjunto de atributos. Um arquivo com a extensão .TXT pode ser lido ou aberto por qualquer programa que leia texto e, com isso, são considerados universais (ou plataforma independente).

É importante lembrar que tais formatos contemplam a especificação de dados para ambos os tipos de aprendizado, supervisionado (descrição da classe do dado de dado incluída) ou não supervisionado (não existe classe associada aos dados).

Os atributos que descrevem os dados devem ser sequenciais, na mesma linha e separados por meio do caractere vírgula (“,”), no caso de dados com a classe associada, esta comparece como um último atributo. Cada atributo do tipo numérico *Real* deve conter o caractere “.” (ponto) separando a parte inteira da fracionária. O restante das informações no arquivo (excluindo os dados) deve ter suas linhas iniciadas pelos caracteres “%” (porcentagem) ou “@” (arroba). As linhas que começam com “%” são comentários e as linhas iniciadas com “@” são as declarações da relação (nome do conjunto de dados). Os dados são descritos logo abaixo da declaração ‘@DATA’. O sistema aceita conjuntos com ou sem a informação da classe. A Figura 5.4 exemplifica o

conteúdo de dois arquivos texto, o primeiro sem a classe associada aos dados (Figura 5.4(a)) e o segundo com a classe associada aos dados (Figura 5.4(b)).

Após o arquivo texto ter sido selecionado, ocorre a sua visualização automática de acordo com o nível de detalhamento definido na aba *Detalhamento* (aba 1). No caso de ocorrer a escolha da opção do nível de detalhamento igual a *Nível* 3 (na aba 1) e de alguma opção de pré-processamento nos dados (na aba 2), a 'planilha' que representa a visualização dos vetores de pares *atributo-valor* presente na aba 2 é atualizada. Para que isso ocorra, a implementação no momento da leitura do arquivo texto (e previamente a sua visualização) faz a criação e o preenchimento de uma matriz que representa os vetores de pares *atributo-valor* dos dados que serão agrupados. No caso do arquivo texto não apresentar a informação da classe, é adicionada ao final da matriz criada uma coluna de atributo para posterior atribuição do número do grupo ao dado. Os valores desta última coluna de atributos da matriz inicialmente é inicializada com zeros (convenção para caracterizar dados sem grupo) e possibilita, após as atribuições dos números dos grupos aos dados, apresentar um agrupamento.

```
% 1. Title: Example Dataset

@RELATION Example1

@ATTRIBUTE a    REAL
@ATTRIBUTE b    REAL
@ATTRIBUTE c    REAL
@ATTRIBUTE d    REAL
@DATA
5.1,3.5,1.4,0.2
4.9,3.0,1.4,0.2
4.7,3.2,1.3,0.2
4.6,3.1,1.5,0.2
.
.
.
```

(a)

```
% 1. Title: Example Dataset

@RELATION Example1

@ATTRIBUTE a    REAL
@ATTRIBUTE b    REAL
@ATTRIBUTE c    REAL
@ATTRIBUTE d    REAL
@ATTRIBUTE class {e,f,g}

@DATA
5.1,3.5,1.4,0.2,a
4.9,3.0,1.4,0.2,a
4.7,3.2,1.3,0.2,c
4.6,3.1,1.5,0.2,b
.
.
.
```

(b)

Figura 5.4 Exemplo de um arquivo texto compreendido pelo sistema. (a) Dados sem a classe associada. (b) Dados com a classe associada.

Além da matriz de vetores de pares *atributo-valor* (ou as matrizes, uma para o agrupamento e outra para o agrupamento com refinamento) necessária(s), a implementação também cuida de criar dois vetores associados (a cada matriz, se for o caso), em que um é para armazenar o centróide de cada um dos grupos que participa do agrupamento e outro é para alocar a quantidade de dados associados a cada um deles.

A Figura 5.5 ilustra as três estruturas de dados (para o caso de apenas uma matriz criada). A Figura 5.5(a) representa a matriz de vetores (a coluna G é inicializada com zeros e será preenchida com o valor correspondente ao grupo). A Figura 5.5(b) corresponde ao vetor dos centróides que indica através dos seus índices (o índice zero não é utilizado) os grupos criados *(1, 2, ..., q)* e o elemento preenchido em uma dessas posições (índices no vetor) indica o índice do centróide do grupo na matriz. A Figura 5.5(c) mostra o vetor que contém as quantidades de dados em cada grupo (incluindo a quantidade de dados não alocados a grupo indicada na posição zero do vetor).

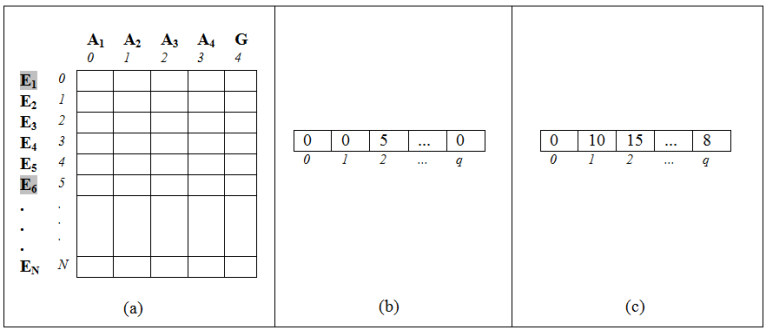


Figura 5.5 (a) Matriz que representa os vetores de pares *atributo-valor*, (b) Vetor de centróides de grupo e (c) Vetor que armazena a quantidades de dados de cada grupo.

Na Figura 5.5(a) estão sombreados os dados de entrada E_1 e E_6 definidos como centróides do grupo 1 e grupo 2, respectivamente. Dessa forma, a Figura 5.5(b) armazena na posição correspondente a cada grupo (1, 2, ..., q) o índice de entrada na matriz do respectivo representante, ou seja, a posição 1 do vetor de centróides (centróide do grupo 1) está preenchido com o valor 0 (zero) que corresponde ao índice de entrada do centróide (E_1) na matriz, o mesmo ocorre com o centróide do grupo 2 que está no índice de entrada 5 da matriz (E_6) e indicado na posição 2 do vetor de representantes. Semelhante ao vetor de centróides, o vetor da Figura 5.5(c) armazena na posição de cada grupo (0, 1, 2, ..., q) a quantidade de dados atribuídos, nesse caso o índice zero desse vetor é utilizado e armazena a quantidade de dados não alocados a nenhum grupo, ou seja, conforme o exemplo nenhum (zero) dado está sem grupo, 10 dados no grupo 1, 15 dados no grupo 2 e 8 dados no último grupo criado (q).

A aba *Algoritmo de Agrupamento* (aba 3), exibida na Figura 5.6, permite ao usuário selecionar um algoritmo de agrupamento, um procedimento de refinamento e um de validação. As opções de algoritmos de agrupamento implementados e

disponibilizados na aba 3 são: BSAS, MBSAS, TTSAS e K-Means, como discutidos no Capítulo 3. O usuário, além de selecionar o algoritmo de interesse, deve também informar os valores de parâmetros relacionados ao algoritmo de escolha, para viabilizar sua execução. Dentre os parâmetros estão: (a) número máximo de grupos (*q*); limiar *Threshold 1*; limiar *Threshold* 2; e concordância (ou não), para o caso do TTSAS, com o processo de atualização de centróide (uma função que atualiza os valores dos atributos do centróide de um grupo a cada atribuição de uma novo dado). Obviamente, os valores de parâmetros a serem informados pelo usuário dependem do algoritmo por ele escolhido:

(1) no caso de ter sido o BSAS ou MBSAS, o SEQ_CLUSTER habilita os campos de número máximo de grupos (*q*) e o limiar (*Threshold 1*);

(2) com a escolha do TTSAS, são habitados os campos de dois limiares (*Threshold 1* e *Threshold 2*) e a opção para atualizar centróide. Essa opção não faz parte do algoritmo e pseudocódigo do TTSAS descritos no Capítulo 3. Ela foi introduzida motivada pelo estudo e implementação realizados durante o trabalho descrito neste material. A ideia que a subsidia é a de que se a opção for escolhida para a atualização do centróide, a cada novo dado atribuído a um determinado grupo os valores dos atributos do centróide daquele grupo serão atualizados, como ocorre com o BSAS e MBSAS.

Na aba 3 estão também disponibilizadas, exceto para o K-Means, as opções de uso dos procedimentos de refinamento implementados: *merge*, *reassignment* e *merge+reassignment*. Além de escolher a opção, o usuário deve também informar o valor do parâmetro esperado pelo procedimento *merge* (analogamente para o *merge+reassignment*) denominado '*Close*', que permite identificar em um agrupamento, os grupos que estão próximos o suficiente para serem juntados. Como já visto, se uma dessas opções de refinamento for escolhida, a implementação cria e preenche outra 'matriz' de vetores que representa o agrupamento aplicado a algum método de refinamento, ou seja, o sistema mantém duas matrizes de dados, uma que representa apenas o agrupamento gerado pelo algoritmo de agrupamento (seja ele BSAS, MBSAS ou TTSAS) e outra que representa tal agrupamento após ter sido tratado

por um dos refinamentos: *merge*, *reassignment* ou *merge+reassignment*. Com isso é possível a visualização dos relatórios gerados.

Na aba 3 estão ainda disponibilizadas ferramentas de validação do agrupamento gerado, através de validação externa e de dois índices de validação interna:

(1) externa: quantifica o número de dados atribuídos corretamente (usando a classe original que faz parte da descrição de cada dado). Esta opção é habilitada apenas para os conjuntos de dados que apresentam a informação da classe.

(2) índice de Dunn (validação interna): que busca identificar em um agrupamento, grupos compactos e bem separados (quanto mais alto o valor deste índice, melhor é a configuração de grupos obtida).

(3) índice de Davies Bouldin (validação interna): que se baseia na medida de dispersão de um grupo e na medida de dissimilaridade entre grupos (quanto mais baixo os valores deste índice, melhor é a configuração de grupos obtida).

Após as definições e eventuais tratamentos dos dados a serem agrupados, bem como do fornecimento dos valores de parâmetros necessários, os algoritmos de agrupamento são executados por meio do botão '*iniciar*' (aba 3). Como visto, a implementação faz com que, em caso de um corrente dado ter sido atribuído a algum grupo (para agrupamento sem ou com processo refinamento), a última coluna da matriz de vetores correspondente é preenchida com a identificação numérica do grupo ao qual foi atribuído, caso contrário permanece com o valor zero indicando que o dado não foi alocado a nenhum grupo.

Os relatórios dos resultados gerados ficam disponíveis conforme os esquemas de experimentos realizados, que são: (1) apenas do algoritmo de agrupamento, (2) algoritmo de agrupamento com refinamento, (3) algoritmo de agrupamento e validação e (4) algoritmo de agrupamento com refinamento e validação. Um exemplo gerado nessa etapa (*Método de Agrupamento*) é mostrado na Figura 5.6 utilizando o esquema: algoritmo BSAS, refinamento *reassignment* e validação externa. O conjunto de dados é o referente à planta Íris (ver Capítulo 1, Figura 1.1).

No exemplo em questão o parâmetro referente ao número de grupos foi estabelecido como 3 e o valor do parâmetro Threshold1 foi estabelecido como 2 (como podem ser visualizados na Figura 5.6).

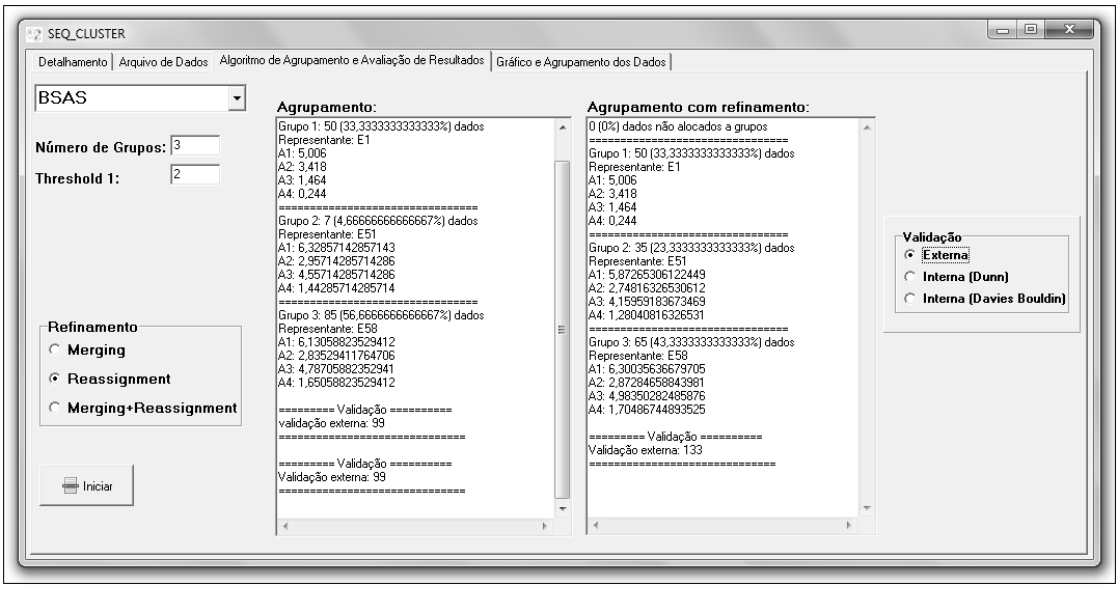


Figura 5.6 Exemplo de agrupamentos (com e sem o refinamento *reassignment*) gerados pelo BSAS, e respectivos relatórios, bem como resultados da validação externa para cada um.

O SEQ_CLUSTER também disponibiliza a opção de visualização do agrupamento por meio da representação gráfica via plotagem das coordenadas dos pontos de dados. A Figura 5.7 mostra a aba *Gráfico e Agrupamento dos Dados* em que é possível gerar e visualizar tal representação e também, a informação nos dados com o grupo ao qual cada um pertence.

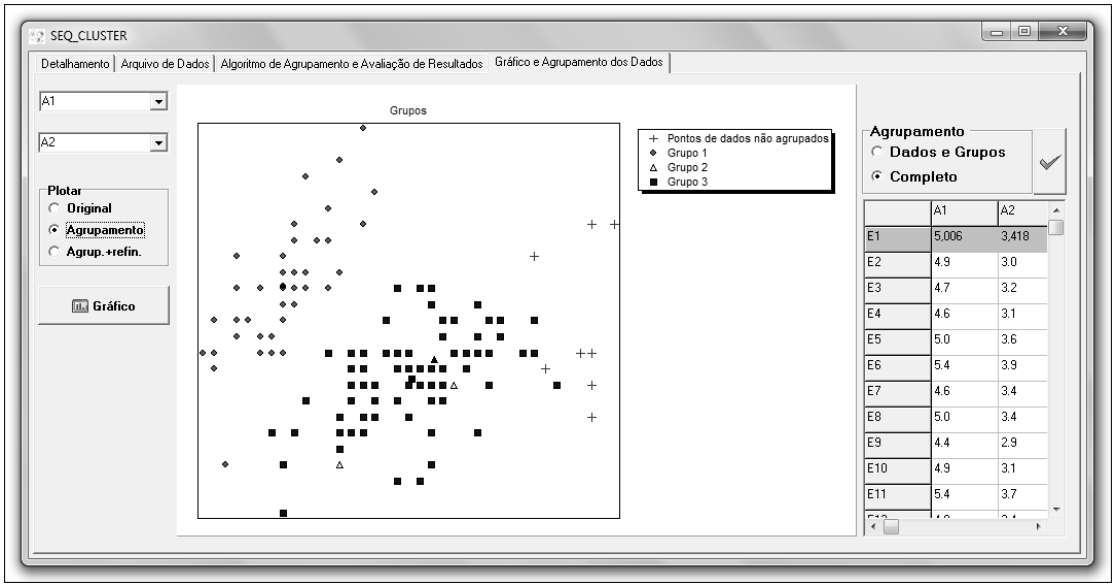


Figura 5.7 Visualização do agrupamento por meio de gráfico.

A representação gráfica planar pode ser gerada a partir de dois atributos do conjunto de M atributos $\{A_1, A_2, ..., A_M\}$ de dados. Cada dado é representado por um ponto no gráfico. Os pontos que compõem os grupos são diferenciados por objetos com formas diferentes, por exemplo, círculo, triângulo, retângulo, estrela, diamante (losango), entre outros. Cada grupo tem seu formato indicado na legenda fornecida. Para a representação gráfica é permitido ao usuário escolher a maneira de como quer visualizar o conjunto de dados, por meio de uma dentre três opções:

(1) *original*: que é a representação gráfica do conjunto de dados original, i.e., sem passar por um processo de agrupamento por meio dos algoritmos;

(2) *agrupamento*: que é a representação gráfica do agrupamento gerado por um dos algoritmos;

(3) *agrup.+refin.*: que é a representação gráfica do agrupamento gerado após aplicado algum dos refinamentos.

Na visualização dos dados é permitido ao usuário escolher o detalhamento de como quer visualizar os dados rotulados, por meio de duas opções em *Agrupamento*:

(1) *Dados e Grupo*, que mostra apenas os *N* dados $\{E_1, E_2, ..., E_N\}$ e a identificação do número do grupo atribuído a cada um;

(2) *Completo*, que mostra cada dado com seu conjunto de atributos, além da identificação do grupo à qual foi atribuída. Nas duas opções os centróides dos grupos estão sombreados.

5.2 Considerações Finais

Este capítulo apresentou o sistema computacional SEQ_CLUSTER que implementa quatro métodos de agrupamento, dois de refinamento (bem como de uma combinação dos dois) e três de validação de agrupamento. Isso possibilita a experimentação dos esquemas propostos em dados reais ou sintéticos, bem como análise de resultados considerando diversas combinações de métodos e condições: pré-processamento, com/sem refinamento como um processo de pós-agrupamento, com/sem validação de agrupamento, influência da ordem dos dados, influência dos parâmetros fornecidos pelo sistema, etc. O Capítulo 6 a seguir apresenta os resultados de inúmeros experimentos conduzidos com os diversos algoritmos sequenciais, considerando possíveis combinações dos métodos, estratégias e condições (parametrização).

Capítulo 6. Experimentos e Análise dos Resultados

Este capítulo descreve os experimentos realizados e as análises dos resultados obtidos com os vários algoritmos de agrupamento sequencial abordados na pesquisa. Nos experimentos foram utilizados tanto dados disponibilizados junto ao repositório do UCI (UCI Repository 2013) quanto sintéticos (i.e., artificialmente criados com foco em conjunto de dados cujos grupos fossem visualmente identificáveis por seres humanos). Todos os experimentos foram realizados utilizando o ambiente computacional viabilizado pelo sistema SEQ_CLUSTER, apresentado e detalhado no Capítulo 5.

Os experimentos foram realizados com o propósito, também, de investigar o desempenho dos algoritmos, de seus refinamentos e de diferentes possibilidades de esquemas de resultados e de condições de entrada (valores de parâmetro, conjunto de dados, etc.).

6.1 Uma Breve Descrição dos Conjuntos de Dados Utilizados nos Experimentos

Na Subseção 6.1.1 os seis conjuntos de dados escolhidos do UCI Repository são brevemente descritos e na Subseção 6.1.2 são apresentados os quatro conjuntos de dados sintéticos que foram criados para a experimentação.

6.1.1 Conjuntos de Dados Utilizados nos Experimentos Extraídos do UCI Repository

Os seis conjuntos de dados extraídos do UCI Repository são brevemente descritos a seguir.

(A) CONJUNTO IRIS

O conjunto de dados da Planta *Iris* está disponível no link: http://archive.ics.uci.edu/ml/datasets/Iris. Os pontos de dados participantes do conjunto são formados por quatro atributos numéricos e estão distribuídos em três classes: *Iris Setosa*, *Iris Versicolor* e *Iris Virginica*. A Tabela 6.1 apresenta um resumo das suas principais características. Os dados estão completos i.e., não comparecem pontos de dados que tenham valor de atributo(s) ausente(s).

Tabela 6.1 Resumo do conjunto de dados *Iris*. #NI: número total de pontos de dados, #NA: número de atributos, Atributos: descrição dos atributos, #NC: número de classes e #NI/Classe: número de pontos de dados por classe.

#NI	#NA	Atributos	#NC	#NI/Classe
150	4	Comprimento da sépala (cm) Largura da sépala (cm) Comprimento da pétala (cm) Largura da pétala (cm)	3	50/setosa 50/versicolor 50/virginica

(B) HEART

O conjunto de dados *Heart* está disponível no link: http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/heart/. O conjunto descreve registros de informações relacionadas a possíveis candidatos a exibirem problema cardíaco. O arquivo original de dados é composto por 270 pontos de dados, descritos por 13 atributos, cada um deles associado a uma possível classe: 1 (ausência) ou 2 (presença). A Tabela 6.2 apresenta um resumo das suas principais características.

O arquivo de dados *Heart* disponível com 13 atributos foi extraído de outro arquivo de dados descrito por um conjunto de 75 atributos. Ainda assim, para o trabalho descrito nesta pesquisa, devido aos tipos de atributos que descrevem alguns pontos de dados, tal como os do tipo Nominal, foi adotado para os experimentos como conjunto de atributos que descrevem os pontos de dados os seguintes atributos do conjunto original de 13 atributos: (1) idade, (4) pressão arterial, (5) colesterol (mg/ml), (8) máxima pulsação, (10) oldpeak e (12) número de artérias principais coloridas por fluorescência.

Tabela 6.2 Resumo do conjunto de dados *heart*. #NI: número de pontos de dados, #NA: número de atributos utilizados do conjunto original, Atributos: descrição dos atributos utilizados do conjunto original, #NC: número de classes e #NI/Classe: número de instâncias por classe.

#NI	#NA	Atributos	#NC	#NI/Classe
270	6	(1) idade (real) (4) pressão arterial em repouso (real) (5) colesterol (mg/dl) (real) (8) máxima pulsação (real) (10) oldpeak (real) (12) número de artérias principais coloridas por fluorescência (real)	2	120/0 150/1

(C) CONJUNTO E.COLI

O conjunto de dados da bactéria *E.coli* está disponível no link: http://archive.ics.uci.edu/ml/datasets/Ecoli. O conjunto possui 336 pontos de dados, cada um deles descrito por valores associados a sete atributos distribuídos em oito classes. A classe indica a localização de uma proteína no organismo do *E.coli*: (1) cp (citoplasma), (2) im (membrana interna sem sinal de sequência), (3) pp (periplasma), (4) imU (membrana interna, sinal de sequência sem clivagem), (5) om (membrana externa), (6) omL (lipoproteína de membrana externa), (7) imL (lipoproteína de membrana interna) e (8) imS (membrana interna, sinal de sequência com clivagem).

É importante mencionar que cada ponto de dado no conjunto original disponibilizado é, de fato, descrito por oito atributos numéricos e que o primeiro atributo que comparece na descrição (i.e. o Sequence Name: Accession number for the SWISS-PROT database) é um identificador junto a um banco de dados e, por essa razão, não comparece na descrição dos dados referentes ao *E.coli* utilizados neste trabalho).

A Tabela 6.3 apresenta um resumo das principais informações associadas ao *E.coli*. Valores associados aos sete atributos utilizados são números reais. Os dados estão completos i.e., não comparecem pontos de dados que tenham valor de atributo(s) ausente(s).

Tabela 6.3 Resumo do conjunto de dados *E.coli*. #NI: número de pontos de dados, #NA: número de atributos, Atributos: descrição dos atributos, #NC: número de classes e #NI/Classe: número de instâncias por classe.

#NI	#NA	Atributos	#NC	#NI/Classe
336	7	(2) mcg (reconhecimento de sinal de sequência via McGeoch (real) (3) GVH (reconhecimento de sinal de sequência via von Heijne (real) (4) lip ((binário) (5) Var: Presença de carga em N-terminais previstos de lipoproteínas (binário) (6) aac: pontuação da análise discriminante do conteúdo de aminoácidos e proteínas da membrana externa e periplasmica (real) (7) ALM1: resultados de programa (real) (8) ALM2: resultado de programa (real)	8	143/cp 77/im 52/pp 20/om 5/omL 2/imL 2/imS 35/imU

(D) CONJUNTO SEEDS

O conjunto de dados Seeds está disponível em: http://archive.ics.uci.edu/ml/datasets/seeds. Este é um conjunto de dados com as medições das propriedades geométricas do núcleo pertencentes a três diferentes variedades de trigo. O conjunto tem 210 pontos de dados, cada um descrito por 7 valores de atributos: (1) área A, (2) perímetro P, (3) compactação 4*pi*A/P^2, (4) tamanho do núcleo, (5) largura do núcleo, (6) coeficiente de assimetria e (7) tamanho do sulco do núcleo. Esses dados estão distribuídos em três classes (i.e., 1, 2 e 3). Os dados estão completos i.e., não comparecem pontos de dados que tenham valor de atributo(s) ausente(s).

(E) CONJUNTO WDBC (Wisconsin Diagnostic Breast Cancer)

O conjunto de dados Wdbc está disponível em: http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/. Este é um conjunto de dados para o diagnóstico de câncer de mama. O conjunto possui 596 pontos de dados, cada um deles descrito por valores associados a trinta atributos distribuídos em duas classes que indica o diagnóstico: M = malígno com 212 pontos de dados e B = benígno com 357 pontos de dados.

É importante mencionar que cada ponto de dado no conjunto original disponibilizado é, de fato, descrito por trinta e dois atributos, sendo o primeiro atributo que comparece na descrição um identificador do dado e o segundo atributo, sua classe.

(F) BREAST TUMOR

O conjunto de dados *Breast Tumor* está disponível em: http://archive.ics.uci.edu/ml/datasets/Breast+Tissue. Este é um conjunto de dados com medições de impedância elétrica em amostras de tecido recentemente retirados da mama. O conjunto tem 106 pontos de dados, cada um descrito por 9 valores de atributos e estão distribuídos em seis classes: Car (Carcinoma), Fad (Fibro-adenoma), Mas (Mastopathy), Gla (Glandular), Con (Connective) e Adi (Adipose).

A Tabela 6.4 apresenta um resumo das suas principais características. Os dados estão completos i.e., não comparecem pontos de dados que tenham valor de atributo(s) ausente(s).

Tabela 6.4 Resumo do conjunto de dados *Breast*. #NI: número pontos de dados, #NA: número de atributos, Atributos: descrição dos atributos, #NC: número de classes e #NI/Classes: número de instâncias por classe.

#NI	#NA	Atributos	#NC	#NI/Classe
106	9	(1) Impedância na freqüência zero (2) PA500 (fase do ângulo em 500 KHz) (3) HFS (inclinação de alta freqüência de fase do ângulo) (4) DA (distância de impedância entre as extremidades do espectro) (5) Área (Área sob espectro) (6) A/DA (área normalizada por DA) (7) MAX IP (máximo do espectro) (8) DR (distância entre I0 e a parte real do ponto de frequência máxima) (9) P (comprimento da curva espectral)	6	21/Car 15/Fad 18/Mas 16/Gla 14/Com 22/Adi

6.1.2 Conjuntos de Dados Artificialmente Gerados

Os conjuntos de dados Sintético1a, Sintético1b, Sintético2 e Sintético3 foram criados artificialmente por meio do módulo '*gerador de conjunto de dados*' do sistema computacional SEQ_CLUSTER, usando como critério a fácil identificação visual dos grupos. Com o objetivo de também avaliar a influência do ruído (i.e. atributos com valores ausentes) por meio dos índices de validação, foi gerado o conjunto de dados Sintético1b. Os conjuntos de dados Sintético1a e Sintético1b possuem a mesma estrutura e pontos de dados, porém no segundo foram adicionados valores ausentes de atributos em 10% dos seus pontos de dados. A Tabela 6.5 apresenta um resumo das principais características dos conjuntos de dados sintéticos.

Tabela 6.5 Resumo dos 4 conjunto de dados sintéticos. #NI: número de instâncias de dados, #NC: número de classes e #NI/Classes: número de instâncias por classe. Cada conjunto de dados é formado por dois atributos.

Conjuntos de dados	#NI	#NC	#NI/Classe
Sintético1a e Sintético1b	300	5	60/a 60/b 60/c 60/d 60/e
Sintético2	125	3	40/a 35/b 50/c
Sintético3	250	5	50/a 50/b 50/c 50/d 50/e

O Sintético1a (Figura 6.1(a)) é um conjunto de dados com cinco grupos bem separados. Nesse caso os índices de validação Dunn e Davies-Bouldin utilizados podem aumentar ou diminuir à medida que o número de grupos se altera. A proposta aqui é a de também verificar, além da capacidade de os algoritmos investigados realizarem bons agrupamentos, os principais fatores (discutidos no Capítulo 3) que podem influenciar nos resultados (sensibilidade quanto a ordem de apresentação dos pontos de dados e os parâmetros fornecidos de entrada). Dessa forma, para os experimentos realizados são mostrados apenas os resultados obtidos com a quantidade exata de grupos para os conjuntos.

Para verificar a influência que os valores de atributos ausentes podem causar nos resultados (ver Capítulo 4), foi criado o Sintético1b (Figura 6.1(b)) em que foram gerados pontos de dados com esse problema.

O Sintético2 (Figura 6.2) é um conjunto de dados criado com o objetivo de avaliar o desempenho dos algoritmos (e dos esquemas de resultados) em um conjunto de dados em que grupos não estão bem definidos. Como será visto, essas características podem influenciar o desempenho das medidas de validação externa e interna.

O Sintético3 (Figura 6.3) é um conjunto de dados que contém cinco grupos, e a dificuldade aos algoritmos é que quatro deles podem formar dois pares de grupos (já que estão muito próximos).

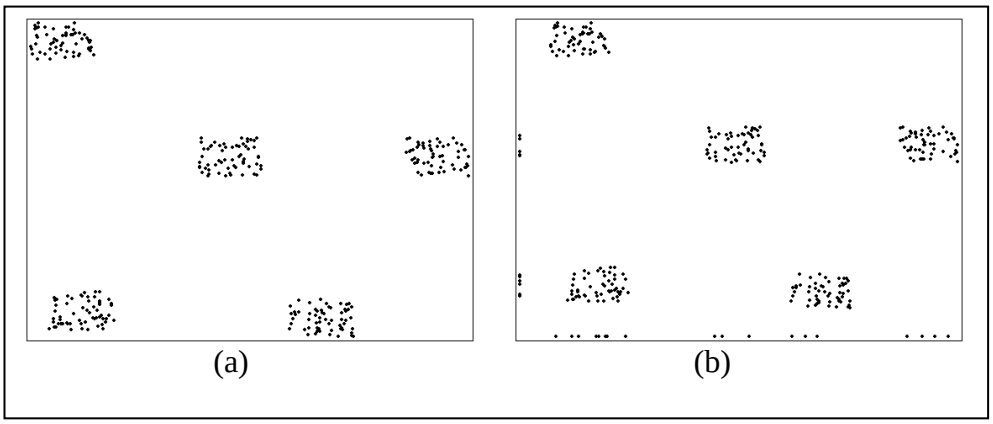


Figura 6.1 O conjunto de dados Sintético1a e Sintético1b. (a) Sintético1a sem valores ausentes. (b) Sintético1b com 10% dos pontos de dados com valores de atributos ausentes.

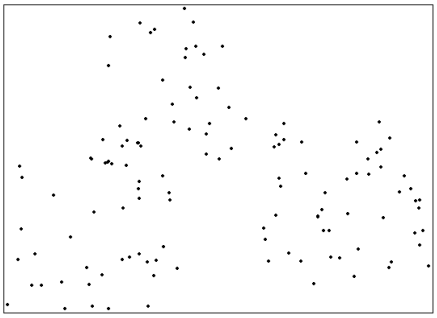


Figura 6.2 O conjunto de dados Sintético2.

Figura 6.3 O conjunto de dados Sintético3.

6.2 Descrição dos Procedimentos Utilizados para os Experimentos

Como comentado anteriormente, nos experimentos foram utilizados um total de seis conjuntos de dados extraídos do UCI Repository e quatro conjuntos de dados sintéticos (artificialmente criados). Os conjuntos de dados têm um número variável de

pontos de dados, todos descritos por atributos numéricos. Para as experiências de agrupamento a distância Euclidiana foi usada para medir a dissimilaridade.

Cada conjunto de dados foi a entrada para cada um dos algoritmos de agrupamento, ou seja, BSAS, MBSAS, TTSAS e K-Means. Para cada conjunto de dados e para cada algoritmo de agrupamento (exceto para o K-Means), quatro resultados foram obtidos, tendo em conta quatro possíveis esquemas levando em consideração estratégias de refinamento:

(1) sem refinamento (SR),

(2) usando apenas *merge* (M) (Algoritmo 3.7),

(3) usando apenas *reassignment* (R) (Algoritmo 3.8) e

(4) utilizando ambos, *merge* e *reassignment* (MR).

Para cada resultado obtido, foram consideradas três validações do agrupamento final: (1) validação externa, (2) índice de Dunn e (3) indice de Davies-Bouldin. Para o K-Means são apresentados apenas os resultados relativos à opção sem refinamento (SR) e a validação externa (VE).

Na obtenção de cada resultado, para cada conjunto de dados, foram realizadas doze execuções do algoritmo escolhido, ou seja, em cada um dos quatro possíveis esquemas de resultados foram aplicadas três validações de agrupamento (dado que cada esquema e cada validação são opções exclusivas no sistema). Este número de execuções pode triplicar (para trinta e seis) no caso da necessidade de ocorrer pré-processamento, visto que, para fins de comparação, podem ser utilizadas duas opções para este processo, além dos resultados obtidos também sem processamento.

Os resultados estão organizados em tabelas associadas a cada conjunto de dados (nas respectivas subseções deste capítulo), conforme a opção de validação de agrupamento utilizada: Validação Externa (VE), Índice de Dunn (D) e Índice de Davies-Bouldin (DB). Os valores dos resultados nas tabelas correspondem aos resultados obtidos pelos três algoritmos, BSAS, MBSAS e TTSAS, respectivamente, tendo em conta os quatro esquemas, SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment* (além do K-Means gerado apenas SR). Os valores destacados em negrito representam os melhores valores obtidos de acordo com a validação de

agrupamento utilizada, i.e., correspondem às porcentagens mais baixas para VE, os valores de índices mais altos para D e os valores de índices mais baixos para DB (isto não exclui que outros bons valores também são encontrados nas tabelas, sendo que muitos deles são tão bons quanto os valores destacados em negrito).

Para viabilizar o esquema, cada conjunto de dados teve seus pontos de dados embaralhados de maneira a mudar sua posição no conjunto; o processo foi repetido nove vezes dando origem a dez conjuntos de dados com exatamente os mesmos pontos de dados, mas diferentes entre si na ordem em que tais pontos comparecem no conjunto (por exemplo, para o *Iris* os conjuntos são denominados Iris1, Iris2, ..., Iris10, para o *Heart* são denominados Heart1, Heart2, ..., Heart10, e assim por diante).

A Figura 6.4 ilustra o processo. Dessa forma, os resultados finais para VE e índices D e DB foram obtidos individualmente para cada conjunto de dados de cada domínio e também informados como a média dos resultados sob um esquema de repetição (dez vezes). Os valores da VE são representados pela % correspondente aos resultados incorretos da comparação do grupo ao qual o dado foi alocado e da classe a qual o dado pertence originalmente, ou seja, para a VE é necessário que a informação classe esteja associada ao dado. Os valores dos parâmetros de entrada foram definidos por meio de tentativas, as quais permitissem encontrar valores para serem utilizados como referência (ou próximo disso). No entanto tais valores não garantem bons agrupamentos e não podem ser definidos como os valores exatos de referência, ou seja, tentativas com diferentes valores podem ser utilizadas a fim de encontrar melhores agrupamentos para os dados.

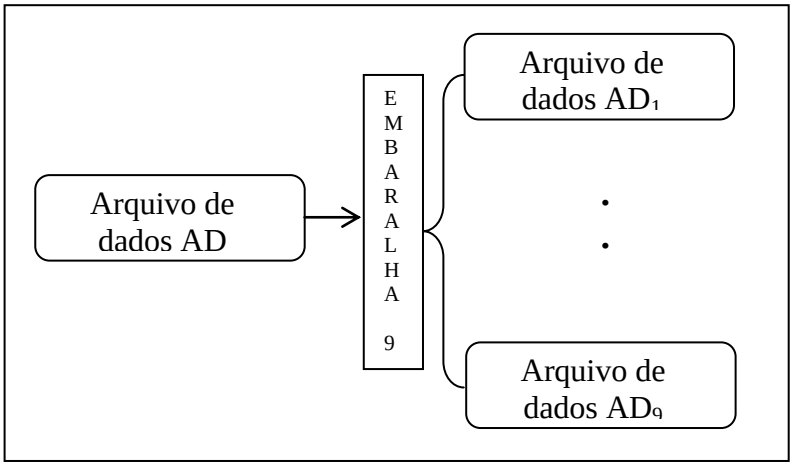


Figura 6.4 Esquema de 'embaralhamento' dos pontos de dados nos conjuntos utilizados nos experimentos.

6.3 Experimentos e Análises de Resultados por Domínio

Esta seção apresenta os resultados e as análises dos experimentos obtidos em cada domínio (conjunto de dados) descrito na Seção 6.1.

6.3.1 IRIS

Nas Tabelas 6.6 a 6.14 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 3, Θ_1: 2,5 e *Close*: 1; e para o terceiro algoritmo foram: (1) Θ_1: 2,5, Θ_2: 5 e *Close*: 2.

Os valores da VE mostram que, usando o BSAS e MBSAS bons resultados podem ser obtidos e, na maioria dos conjuntos, os valores podem ser melhorados com a utilização de algum método de refinamento.

No BSAS (Tabela 6.6), com exceção do Iris7, em todos os conjuntos o algoritmo obteve um bom desempenho, com destaque para o Iris6 que alocou corretamente todos os pontos de dados utilizando qualquer dos esquemas. Já para os conjuntos Iris8 e Iris10 a utilização de um método de refinamento (R ou MR) foi fundamental para a melhoria dos seus agrupamentos.

O MBSAS (Tabela 6.7) teve o seu desempenho um pouco inferior ao BSAS, no entanto alguns dos valores obtidos podem ser considerados muito bons (veja Iris5, Iris7, Iris8 e Iris10). O conjunto Iris4 é o que foi melhor agrupado apresentando o melhor desempenho com todos os seus pontos de dados alocados corretamente. Ainda com relação ao MBSAS, o uso de um processo de refinamento não influenciou nos resultados (este tipo de comportamento é influenciado, geralmente, de acordo com os valores de parâmetros utilizados).

Com relação ao TTSAS (Tabela 6.8) o seu desempenho foi inferior quando comparado aos desempenhos do BSAS e do MBSAS, devido principalmente à quantidade de grupos criados (geralmente maior que três). Entretanto essa situação pode ser melhorada com o fornecimento de valores de parâmetros de entrada diferentes dos que foram utilizados no experimento (i.e. Iris8, que obteve o pior resultado da VE no TTSAS utilizando Θ_1: 2,5, Θ_2: 5 e *Close*: 2, se for utilizado Θ_1: 1,5, Θ_2: 3 e *Close*: 1 consegue valores melhores de VE: SR=22%, M=22%, R=40% e MR=40%). Mesmo

com a influência dos valores dos parâmetros e um desempenho inferior comparado ao BSAS e ao MBSAS, a maioria dos resultados usando o TTSAS está abaixo dos 50% de dados alocados incorretamente – com exceção dos conjuntos Iris9 e Iris10 – todos os outros apresentaram um melhor desempenho quando do uso de refinamento.

A Tabela 6.9 apresenta um resumo com as médias dos resultados da VE utilizando o BSAS, MBSAS e TTSAS, considerando os quatro esquemas. No K-Means (Tabela 6.10), muitos conjuntos apresentaram resultados bem próximos comparados aos três algoritmos, porém ainda assim o BSAS, MBSAS e o TTSAS foram mais eficientes. No BSAS apenas os valores para conjunto Iris7 ficaram inferior aos do K-Means (exceto quando utilizado o refinamento R).

Tabela 6.6 VE do BSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Iris1	3,33	3,33	9,33	9,33
Iris2	3,33	33,33	3,33	3,33
Iris3	13,33	43,33	34,00	23,33
Iris4	23,33	43,33	38,00	23,33
Iris5	23,33	43,33	35,33	23,33
Iris6	**0,00**	**0,00**	**0,00**	**0,00**
Iris7	54,67	62,00	31,33	62,00
Iris8	40,67	40,67	28,67	28,67
Iris9	12,67	12,67	6,67	6,67
Iris10	59,33	66,67	32,67	32,67

Tabela 6.7 VE do MBSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Iris1	47,33	47,33	54,67	54,67
Iris2	37,33	37,33	44,67	44,67
Iris3	37,33	37,33	44,67	44,67
Iris4	**0,00**	**0,00**	**0,00**	**0,00**
Iris5	1,33	1,33	0,67	0,67
Iris6	41,33	41,33	48,67	48,67
Iris7	4,00	4,00	6,00	6,00
Iris8	8,67	8,67	10,67	10,67
Iris9	47,33	47,33	54,67	54,67
Iris10	12,67	12,67	10,00	10,00

Tabela 6.8 VE do TTSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Iris1	66,67	36,00	58,67	44,67
Iris2	56,67	26,00	48,67	34,67
Iris3	56,67	**23,33**	49,33	33,33
Iris4	50,67	**23,33**	47,33	**23,33**
Iris5	45,33	45,33	34,00	34,00
Iris6	60,67	30,00	57,33	38,67
Iris7	61,33	61,33	52,00	52,00
Iris8	60,67	60,67	56,67	56,67
Iris9	55,33	55,33	44,67	44,67
Iris10	56,67	56,67	44,67	44,67

Tabela 6.9 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	23,40	34,87	21,93	**21,27**
MBSAS	23,73	23,73	27,47	27,47
TTSAS	57,07	41,80	49,33	40,67

Tabela 6.10 VE do K-MEANS para cada um dos conjuntos de dados *Iris*.

Conjuntos de dados	SR
Iris1	52,67
Iris2	46,00
Iris3	76,00
Iris4	76,00
Iris5	**45,33**
Iris6	50,00
Iris7	50,00
Iris8	56,00
Iris9	76,00
Iris10	55,33

Considerados os valores de validação interna obtidos usando os índices D e DB é possível notar que alguns conjuntos apresentam melhores estruturas de agrupamentos que outros. No caso do BSAS (Tabela 6.11), o conjunto Iris1 foi o que obteve uma melhor configuração quando gerado pelas combinações SR, M e R (ou seja, obteve um índice D com um valor maior que os demais conjuntos e um índice DB com um baixo valor).

Com relação aos resultados obtidos pelo MBSAS (Tabela 6.12) há uma disparidade as avaliações obtidas pelo D e DB. Enquanto que em Iris7 e Iris8 em qualquer dos quatro esquemas o índice D aponta como os de melhor resultado (maiores

valores de D), o índice DB aponta como os de melhor resultado aqueles obtidos com Iris3 (SR e M) e Iris4 (R e MR), ou seja, o menores valores de DB.

Com relação ao TTSAS (Tabela 6.13) também houve inconsistências entre os valores dos índices D e DB. O que pode ter interferido talvez tenha sido a quantidade de grupos criados. Ainda assim os conjuntos Iris1, Iris2, Iris6 e Iris9 apresentaram boas configurações quando utilizados nos esquemas SR e M. A Tabela 6.14 mostra um resumo das médias dos valores dos índices D e DB para os agrupamentos dos dez conjuntos gerados.

Tabela 6.11 D e DB do BSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Iris1	0,05848	0,48664	0,05848	0,48664	0,09091	0,46141	0,09091	0,46141
Iris2	0,03156	2,45645	0,00846	0,27589	0,06395	1,36899	**0,33891**	**0,22793**
Iris3	0,03156	2,04585	0,01316	0,36027	0,03629	0,53473	**0,33891**	0,28633
Iris4	0,01996	1,96237	0,01316	0,36061	0,04530	0,50655	**0,33891**	0,28678
Iris5	0,01996	2,18215	0,01316	0,36345	0,03745	0,52523	**0,33891**	0,28403
Iris6	0,04610	0,50783	0,04610	0,50783	0,08904	0,40828	0,08904	0,40828
Iris7	0,01705	1,10023	0,01705	0,28691	0,07651	0,66393	0,03714	0,22808
Iris8	0,03328	0,86552	0,03328	0,86552	0,04940	0,44105	0,04940	0,44105
Iris9	0,02922	0,66143	0,02922	0,66143	0,07674	0,54348	0,07674	0,54348
Iris10	0,01715	0,96786	0,01715	0,32438	0,03405	0,61592	0,08111	0,24825

Tabela 6.12 D e DB do MBSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento. M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjunto	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Iris1	0,08657	0,35923	0,08657	0,35923	0,06395	0,39836	0,06395	0,39836
Iris2	0,08657	0,35856	0,08657	0,35856	0,06395	0,39766	0,06395	0,39766
Iris3	0,08657	**0,35658**	0,08657	**0,35658**	0,06395	0,39561	0,06395	0,39561
Iris4	0,09881	0,37781	0,09881	0,37781	0,09881	0,37984	0,09881	0,37984
Iris5	0,04727	0,37944	0,04727	0,37944	0,09881	0,38828	0,09881	0,38828
Iris6	0,08657	0,36218	0,08657	0,36218	0,06395	0,40124	0,06395	0,40124
Iris7	0,13783	0,38280	0,13783	0,38280	**0,13973**	0,38838	**0,13973**	0,38838
Iris8	0,13783	0,38518	0,13783	0,38518	**0,13973**	0,39066	**0,13973**	0,39066
Iris9	0,08657	0,35923	0,08657	0,35923	0,06395	0,40329	0,06395	0,40329
Iris10	0,04727	0,38102	0,04727	0,38102	0,13346	0,38622	0,13346	0,38622

Tabela 6.13 D e DB do TTSAS para cada um dos conjuntos de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjunto	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Iris1	0,69393	0,53974	0,63902	0,26851	0,05249	0,58361	0,04522	0,43871
Iris2	0,69393	0,53808	0,63902	**0,26647**	0,05249	0,58384	0,04522	0,43902
Iris3	0,16949	0,61480	0,15604	0,35274	0,04706	0,63662	0,05374	0,48740
Iris4	0,03699	0,62559	0,33891	0,28134	0,07968	0,55864	0,08111	0,28107
Iris5	0,04727	0,55566	0,04727	0,55566	0,07226	0,49223	0,07226	0,49223
Iris6	0,69393	0,53970	0,63902	0,26845	0,05249	0,58518	0,04522	0,44088
Iris7	0,05848	0,48002	0,05848	0,48002	0,11279	0,58518	0,11279	0,44911
Iris8	0,04530	0,42327	0,04530	0,42327	0,07780	0,39704	0,07780	0,39704
Iris9	**0,87956**	0,53970	**0,87956**	0,53970	0,05472	0,52140	0,05472	0,52140
Iris10	0,04688	0,39789	0,04688	0,39789	0,07264	0,36369	0,07264	0,36369

Tabela 6.14 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Iris* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Esquema de	BSAS		MBSAS		TTSAS	
resultado	D	DB	D	DB	D	DB
SR	0,03043	1,32363	0,09018	0,37020	0,33657	0,52544
M	0,02492	0,44929	0,09018	0,37020	0,34895	0,38341
R	0,05996	0,60696	0,09303	0,39295	0,06744	0,53074
MR	**0,17800**	**0,34156**	0,09303	0,39295	0,06607	0,43106

6.3.2 HEART

Nas Tabelas 6.15 a 6.23 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 2, Θ_1: 100 e *Close*: 50; e para o terceiro algoritmo foram: (1) Θ_1: 100, Θ_2: 200 e *Close*: 150.

Os valores da VE mostram que os três algoritmos não conseguiram obter resultados dentro de uma margem aceitável de erros. No entanto, alguns conjuntos de dados tiveram seus dados um pouco melhor alocados quando do uso dos métodos de refinamento.

O MBSAS (Tabela 6.16) foi o que obteve melhor desempenho em comparação com o BSAS (Tabela 6.15) e o TTSAS (Tabela 6.17), e apresenta resultados semelhantes (evidenciando uma certa ‘estabilidade’) em todos os conjuntos de dados gerados em qualquer combinação de resultados (SR, M, R ou MR). A taxa máxima de

50% de pontos de dados alocados incorretamente é encontrada em apenas um dos dez conjuntos (Heart10+R).

Mais uma vez, além da ordem de apresentação dos pontos de dados, fica ratificado que os valores de parâmetros de entrada podem influenciar nos resultados, como no caso do Heart2 que, gerado no BSAS, obteve resultados muito ruins em duas combinações de resultados (SR=90% e M=90%), mas que se utilizado um $\Theta_1 = 150$ (ao invés de $\Theta_1 = 100$) as porcentagens de erros de alocação diminuem para SR=55,9% e M=55,9% (ainda assim não são bons resultados, porém indica um caminho para outras tentativas).

Já se para esse mesmo conjunto de dados (Heart2) for realizada uma pequena alteração na ordem dos dados, o resultado no BSAS utilizando $\Theta_1 = 100$ fica em: SR=56,6%, M=56,6%, R=55,5% e MR=55,5%, já se for utilizado $\Theta_1 = 150$ os resultados melhoram um pouco e serão de: SR=55,9%, M=55,9%, R=55,5% e MR=55,5%.

Na Tabela 6.18 pode ser visto um resumo com as médias dos resultados obtidos em cada conjunto utilizando o BSAS, MBSAS e TTSAS e os quatro esquemas.

Em comparação de desempenho com os resultados gerados pelo K-Means (Tabela 6.19), os três algoritmos (BSAS, MBSAS e TTSAS) foram inferiores em quatro conjuntos (Heart2, Heart3, Heart6 e Heart10), sendo que nos demais os resultados foram satisfatórios principalmente quando utilizados em algum método de refinamento.

Em suma, as estratégias de refinamentos foram fundamentais no Heart para a melhoria dos agrupamentos na maioria dos conjuntos de dados, evidenciado principalmente quando o TTSAS foi usado com os esquemas M e MR. Em geral, no domínio Heart, os quatro algoritmos obtiveram resultados bem semelhantes.

Tabela 6.15 VE do BSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Heart1	56,30	56,30	57,04	57,04
Heart2	90,00	90,00	57,78	57,78
Heart3	56,30	56,30	57,04	57,04
Heart4	57,04	58,15	59,26	**44,44**
Heart5	57,41	47,04	45,19	**44,44**
Heart6	56,67	56,67	57,78	57,78
Heart7	57,04	46,30	57,78	57,78
Heart8	57,04	46,30	49,26	**44,44**
Heart9	56,30	56,30	57,78	57,78
Heart10	55,93	55,93	58,52	58,52

Tabela 6.16 VE do MBSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Heart1	44,81	44,81	44,81	44,81
Heart2	44,81	44,81	44,07	44,07
Heart3	44,81	44,81	44,81	44,81
Heart4	40,74	40,74	**39,63**	**39,63**
Heart5	44,44	44,44	43,33	43,33
Heart6	44,81	44,81	44,81	44,81
Heart7	44,81	44,81	43,70	43,70
Heart8	44,44	44,44	43,70	43,70
Heart9	44,81	44,81	44,07	44,07
Heart10	44,81	44,44	50,00	44,44

Tabela 6.17 VE do TTSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Heart1	74,81	44,81	94,81	45,56
Heart2	74,81	44,81	94,07	44,81
Heart3	74,81	44,81	94,81	45,93
Heart4	47,41	46,30	66,30	48,52
Heart5	52,22	44,81	76,67	**42,96**
Heart6	74,81	44,81	94,44	45,19
Heart7	74,81	44,81	96,30	44,81
Heart8	47,41	46,30	79,63	50,00
Heart9	74,81	44,81	94,81	45,19
Heart10	74,81	44,81	97,04	51,11

Tabela 6.18 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	60,00	56,93	55,74	53,70
MBSAS	44,33	44,30	44,30	**43,74**
TTSAS	67,07	45,11	88,89	46,41

Tabela 6.19 VE do K-MEANS para cada um dos conjuntos de dados *Heart*.

Conjuntos de dados	SR
Heart1	58,89
Heart2	**40,37**
Heart3	41,48
Heart4	58,89
Heart5	59,26
Heart6	40,74
Heart7	59,63
Heart8	59,26
Heart9	59,26
Heart10	41,48

Os índices D e DB apresentam valores bem próximos ou mesmo iguais para alguns conjuntos processados em qualquer dos algoritmos. No BSAS (Tabela 6.20) o índice D apontou melhor configuração de agrupamento para os conjuntos Heart1, Heart2, Heart7 e Heart9 considerando os quatro esquemas. No entanto para outros conjuntos, D também aparece com bons valores em alguns esquemas. Já o índice DB (no BSAS) indica que o conjunto Heart10 obteve melhor configuração de agrupamento. Pode ser observado que em três conjuntos os valores para o M ou MR estão representados pelo caractere '-', que indica que foi gerado apenas um único grupo.

Para o MBSAS (Tabela 6.21) os índices D e DB concordaram em eleger o conjunto Heart6 como o de melhor configuração considerando os quatro esquemas. No entanto, muitos dos valores em outros conjuntos e dependendo do esquema utilizado, estão bem próximos dos valores apresentados pelo Heart6.

No TTSAS (Tabela 6.22) acontece a mesma situação apresentada anteriormente em outros domínios, a de depender da quantidade de grupos criados (em que o ideal seria executá-lo por algumas vezes com diferentes valores de parâmetros de entrada (Θ_1 e Θ_2), e quanto melhores os valores de índices obtidos, mais ideal seria a quantidade de grupos). No entanto, ainda assim é possível destacar os valores apresentados de D para o conjunto Heart10, que o caracteriza com o de melhor configuração entre os demais, considerando os quatro esquemas. A Tabela 6.23 mostra um resumo das médias dos valores dos índices D e DB para os agrupamentos dos conjuntos de dados.

Tabela 6.20 D e DB do BSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Heart1	**0,09484**	0,29145	**0,09484**	0,29145	0,05693	0,59361	0,05693	0,59361
Heart2	**0,09484**	0,29206	**0,09484**	0,29206	0,05693	0,57730	0,05693	0,57730
Heart3	**0,09484**	0,29145	**0,09484**	0,29145	0,05640	0,58412	0,05640	0,58412
Heart4	0,02991	2,11935	-	-	0,02074	0,63141	-	-
Heart5	0,03326	3,09138	-	-	0,01207	0,77946	-	-
Heart6	**0,09484**	0,29124	**0,09484**	0,29124	0,04546	0,58356	0,04546	0,58356
Heart7	**0,09484**	0,29301	**0,09484**	0,29301	0,05693	0,58285	0,05693	0,58285
Heart8	0,04651	0,99356	-	-	0,01729	0,72018	-	-
Heart9	**0,09484**	0,29199	**0,09484**	0,29199	0,05693	0,59348	0,05693	0,59348
Heart10	0,05554	**0,15511**	0,05554	**0,15511**	0,03197	0,44893	0,03197	0,44893

Tabela 6.21 D e DB do MBSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Heart1	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05241	0,23971	0,05241	0,23971
Heart2	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05241	0,23971	0,05241	0,23971
Heart3	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05241	0,23971	0,05241	0,23971
Heart4	0,02951	0,46121	0,02951	0,46121	0,02067	0,49750	0,02067	0,49750
Heart5	0,07627	0,27839	0,07627	0,27839	0,03197	0,43643	0,03197	0,43643
Heart6	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05351	0,19435	0,05351	0,19435
Heart7	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05241	0,23971	0,05241	0,23971
Heart8	0,01318	0,54427	0,01318	0,54427	0,02087	0,60109	0,02087	0,60109
Heart9	**1,08445**	**0,07936**	**1,08445**	**0,07936**	0,05241	0,23971	0,05241	0,23971
Heart10	0,04613	1,25930	-	-	0,01207	0,79124	-	-

Tabela 6.22 D e DB do TTSAS para cada um dos conjuntos de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Heart1	0,77796	0,57797	1,52399	0,29454	0,24378	0,58372	1,47049	0,29768
Heart2	0,77796	0,58322	1,52399	0,29517	0,24378	0,58341	1,47049	0,29713
Heart3	0,77796	0,61130	1,52399	0,30108	0,24846	0,58062	1,49187	0,29614
Heart4	1,07257	0,29593	0,64464	0,22051	0,73947	0,44010	0,70746	0,31307
Heart5	0,79877	0,34267	1,16820	**0,08269**	0,22666	0,57703	0,04044	0,40706
Heart6	0,77796	0,57623	1,52399	0,29456	0,24378	0,58484	1,47049	0,29778
Heart7	0,77796	0,55735	1,05235	0,10055	0,24378	0,58896	1,05235	0,10055
Heart8	1,07257	0,31450	0,64464	0,22874	0,40716	0,51629	0,71321	0,36352
Heart9	0,77796	0,55929	1,05235	0,10055	0,24378	0,58702	1,05235	0,10055
Heart10	**1,82557**	0,92747	1,66721	0,28208	1,75208	0,56786	1,75392	0,28762

Tabela 6.23 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Heart* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: merge+*reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,07	0,81	0,67	0,30	0,94	0,53
M	0,09	0,27	0,74	**0,20**	**1,23**	0,22
R	0,04	0,61	0,04	0,37	0,46	0,56
MR	0,05	0,57	0,04	0,33	1,12	0,28

6.3.3 E.COLI

Nas Tabelas 6.24 a 6.32 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 8, Θ_1: 0,5 e *Close*: 0,3; e para o terceiro algoritmo foram: (1) Θ_1: 0,3, Θ_2: 0,8 e *Close*: 0,5.

Os valores da VE apontam para a maioria dos conjuntos deste domínio um melhor desempenho quando alguma estratégia de refinamento foi utilizada. No BSAS (Tabela 6.24) quatro conjuntos (Ecoli2, Ecoli3, Ecoli4 e Ecoli6) obtiveram os melhores resultados com refinamento R. Nos demais conjuntos, embora o refinamento não tenha colaborado na melhoria dos resultados (no BSAS), os valores ficaram próximo dos gerados pelo algoritmo sem refinamento. Entretanto para o MBSAS (Tabela 6.25) os refinamentos M e MR geraram os mais eficientes resultados em todos os conjuntos (exceto para o Ecoli2 e Ecoli4). Como no MBSAS, o TTSAS (Tabela 6.26) obteve melhores resultados quando utilizados com os refinamentos M e MR em todos os conjuntos.

A influência dos parâmetros de entrada nesse domínio pode ser evidenciada focalizando o Ecoli2 por meio de alterações dos valores desses parâmetros, por exemplo. No Ecoli2, o BSAS com Θ_1 = 0,8 e *Close*: 0,4 (ao invés de Θ_1: 0,5 e *Close*: 0,3), as porcentagens de erros diminuem para SR=49,1%, M=38,3%, R=47,3 e MR=37,2. No MBSAS com Θ_1 = 0,8 e *Close*: 0,4 (ao invés de Θ_1: 0,5 e *Close*: 0,3), as porcentagens de erros diminuem para SR=36,9%, M=36,9%, R=37,2 e MR=37,2.

Na Tabela 6.27 é apresentado um resumo com as médias dos resultados obtidos em cada conjunto utilizando o BSAS, MBSAS e TTSAS e os quatro esquemas.

Comparando os resultados gerados pelo K-Means (Tabela 6.28), os três algoritmos (BSAS, MBSAS e TTSAS) têm um desempenho superior.

Tabela 6.24 VE do BSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Ecoli1	**42,86**	56,25	45,24	59,52
Ecoli2	53,57	67,26	43,75	61,31
Ecoli3	64,29	56,85	47,32	68,15
Ecoli4	75,00	56,85	48,21	74,40
Ecoli5	80,95	57,74	65,18	58,93
Ecoli6	44,05	57,74	43,45	59,23
Ecoli7	43,75	57,44	44,64	59,82
Ecoli8	45,54	55,95	47,92	60,71
Ecoli9	79,46	56,55	63,99	58,63
Ecoli10	43,75	57,44	44,35	58,93

Tabela 6.25 VE do MBSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento. M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Ecoli1	78,57	34,23	72,02	34,23
Ecoli2	90,18	88,39	92,86	91,96
Ecoli3	80,65	50,30	76,49	50,30
Ecoli4	80,95	59,23	76,79	58,04
Ecoli5	72,02	44,05	64,58	44,94
Ecoli6	76,19	34,82	71,13	33,93
Ecoli7	78,57	**31,85**	72,62	33,33
Ecoli8	78,57	34,52	70,54	34,23
Ecoli9	93,45	55,65	82,14	54,76
Ecoli10	80,95	38,10	67,26	33,33

Tabela 6.26 VE do TTSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Ecoli1	79,46	56,55	94,05	55,95
Ecoli2	62,80	40,77	88,39	36,90
Ecoli3	60,71	**32,74**	82,74	35,71
Ecoli4	60,12	39,58	81,55	61,61
Ecoli5	76,79	39,58	93,15	44,94
Ecoli6	79,46	57,14	94,05	61,90
Ecoli7	79,46	56,55	93,75	61,90
Ecoli8	79,46	56,55	94,05	55,95
Ecoli9	79,46	56,55	94,05	58,04
Ecoli10	79,46	57,14	94,35	56,25

Tabela 6.27 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Algoritmo	SR	M	R	MR
BSAS	57,32	58,01	49,40	61,96
MBSAS	81,01	47,11	74,64	**46,90**
TTSAS	73,72	49,32	91,01	52,92

Tabela 6.28 VE do K-MEANS para cada um dos conjuntos de dados *Ecoli*.

Conjuntos de dados	SR
Ecoli1	75,89
Ecoli2	89,88
Ecoli3	72,62
Ecoli4	95,24
Ecoli5	90,18
Ecoli6	**69,35**
Ecoli7	93,15
Ecoli8	86,90
Ecoli9	87,80
Ecoli10	98,51

Em muitos dos conjuntos os índices D e DB apresentaram valores bem próximos. No BSAS (Tabela 6.29) é possível verificar por meio dos valores de D e DB que o conjunto Ecoli9 não foi gerado com boa qualidade em relação aos demais, ou seja, nele são encontrados os menores valores de D e os maiores valores de DB.

Para o MBSAS (Tabela 6.30) não houve valor que destacasse algum conjunto agrupado com melhor ou pior qualidade em relação aos demais. Entre os que apresentaram um desempenho mais satisfatório (no entanto com valores bem próximos dos demais conjuntos), estão o Ecoli2 (índice D) quando usado com refinamento R ou MR, o Ecoli3 (índice D) sem refinamento ou usando o refinamento M e o Ecoli9 (índice DB).

Com relação aos valores de índices D e DB considerando o método TTSAS (Tabela 6.31), pode ser evidenciado que dois conjuntos (Ecoli3 e Ecoli4) apresentaram baixos valores em D em relação aos demais. A Tabela 6.32 apresenta um resumo dos valores das médias para D e DB para os quatro esquemas.

Tabela 6.29 D e DB do BSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reatribuição*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Ecoli1	**0,75046**	1,67006	0,72210	0,83743	0,45034	1,17081	0,45034	0,94051
Ecoli2	0,59251	1,56091	0,62155	0,99927	0,45034	1,12940	0,45034	0,92297
Ecoli3	0,57168	1,52926	0,62747	**0,74277**	0,40175	1,13731	0,54374	0,89571
Ecoli4	0,57168	1,47701	0,62747	**0,74241**	0,55775	1,15456	0,54374	0,90792
Ecoli5	0,06353	2,26642	0,04202	1,11956	0,18154	1,26328	0,11404	0,98129
Ecoli6	0,66718	1,94105	0,64516	0,88885	0,45034	1,37122	0,45034	0,92672
Ecoli7	0,71483	1,64944	0,72102	0,85235	0,45034	1,14057	0,45034	0,92191
Ecoli8	0,08223	1,99687	0,04949	1,28363	0,17962	1,17827	0,06751	0,97802
Ecoli9	0,06717	2,34167	0,04443	0,99162	0,18154	1,28715	0,06751	1,01189
Ecoli10	0,64199	1,63640	0,68163	0,89539	0,45034	1,13000	0,45034	0,91895

Tabela 6.30 D e DB do MBSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: reatribuição e MR: merge+reatribuição.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Ecoli1	0,54171	0,87388	0,54162	0,64487	0,68314	0,84952	0,54171	0,67681
Ecoli2	0,53961	0,83812	0,55576	0,70526	**0,72409**	0,85153	0,71568	0,73808
Ecoli3	0,66574	0,86879	0,66574	0,70739	0,69579	0,90219	0,69579	0,71193
Ecoli4	0,58094	0,87704	0,55775	0,64400	0,69579	0,94620	0,59910	0,67998
Ecoli5	0,36568	0,89765	0,36568	0,70492	0,57857	0,93354	0,57857	0,73194
Ecoli6	0,54171	0,85699	0,54162	0,64509	0,68314	0,85076	0,54171	0,67731
Ecoli7	0,54171	0,84347	0,54171	0,63238	0,68314	0,85077	0,54171	0,67524
Ecoli8	0,53537	0,86929	0,53454	0,63589	0,68314	0,86700	0,54171	0,67716
Ecoli9	0,52186	0,81755	0,52186	**0,62273**	0,68314	0,82817	0,59910	0,68809
Ecoli10	0,53961	0,90392	0,51781	0,64599	0,54171	0,85657	0,54171	0,69849

Tabela 6.31 D e DB do TTSAS para cada um dos conjuntos de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Ecoli1	1,22286	0,68242	0,58901	0,50496	1,35212	0,71326	0,70374	0,69958
Ecoli2	1,46864	0,58459	0,66488	**0,41344**	**1,68352**	0,72004	0,49868	0,58711
Ecoli3	0,93730	0,53424	0,49444	0,47378	0,18887	0,77295	0,46511	0,63674
Ecoli4	0,56231	0,70700	0,63839	0,38629	0,42597	0,72923	0,60659	0,61812
Ecoli5	1,22286	0,60053	0,58901	0,50782	1,42271	0,68884	0,70374	0,68579
Ecoli6	1,22286	0,69583	0,58901	0,51614	1,35212	0,73464	0,70374	0,73855
Ecoli7	1,22286	0,59738	0,58288	0,50176	1,42271	0,68841	0,70374	0,73855
Ecoli8	1,22286	0,68242	0,58901	0,50496	1,35212	0,72426	0,70374	0,69970
Ecoli9	1,22286	0,67647	0,58901	0,50496	1,35212	0,71334	0,70374	0,69351
Ecoli10	1,34144	0,61750	0,58901	0,51604	1,33697	0,71843	0,72736	0,67912

Tabela 6.32 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Ecoli* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,47233	1,80691	0,53739	0,86467	1,16468	0,63784
M	0,47824	0,93533	0,53441	0,65885	0,59147	**0,48301**
R	0,37539	1,19626	0,66516	0,87363	**1,18892**	0,72034
MR	0,35882	0,94059	0,58968	0,69550	0,65201	0,67768

6.3.4 SEEDS

Nas Tabelas 6.33 a 6.41 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 3, Θ_1: 5 e *Close*: 0,5; e para o terceiro algoritmo foram: (1) Θ_1: 3, Θ_2: 6 e *Close*: 0,5.

Os valores da VE apresentam um bom desempenho para a maioria dos conjuntos, com ou sem a utilização de algum método de refinamento, ou seja, as estratégias de refinamento não influenciaram os resultados obtidos. Como já examinados para os três primeiros domínios apresentados, para o Seeds também ficou clara a influência da ordem de apresentação dos dados e dos valores atribuídos aos parâmetros de entrada.

No BSAS (Tabela 6.33) o conjunto Seeds1 obteve um ótimo agrupamento em qualquer um dos quatro esquemas. Os conjuntos Seeds2, Seeds3, Seeds5 e Seeds7 também foram bem agrupados; os resultados obtidos com esses conjuntos foram melhorados com as estratégias de refinamento R e MR. É importante notar que o Seeds6 no BSAS obteve os piores erros de alocação (SR=66,19%, M=66,19%, R=91,90% e MR=91,90%). Neste mesmo conjunto utilizando $\Theta_1 = 6$ (ao invés de $\Theta_1 = 5$) as porcentagens de erros diminuem para SR=38,57%, M=38,57%, R=28,10% e MR=28,10%.

O MBSAS (Tabela 6.34) quando usado com os conjuntos Seeds1, Seeds2, Seeds5, Seeds6 e Seeds7 também teve bom desempenho, que foi ainda melhorado com a utilização dos métodos de refinamento R e MR.

O TTSAS (Tabela 6.35) foi eficiente quando usado com todos os conjuntos (exceto Seeds6), inclusive com a utilização dos métodos M e MR. Em contrapartida, os

três algoritmos produziram resultados insatisfatórios em alguns conjuntos, como o Seeds10 (BSAS), Seeds4, Seeds8 e Seeds9 (MBSAS) e Seeds6 e Seeds10 (TTSAS).

A Tabela 6.36 apresenta um resumo com as médias da VE para os quatro esquemas. Comparando os desempenhos obtidos dessa tabela com os resultados gerados pelo K-Means (Tabela 6.37), os três algoritmos (BSAS, MBSAS e TTSAS) são mais eficientes, exceto para o Seeds5 onde o K-Means mostra uma VE de 11,43%. Como visto, o conjunto Seeds5 foi o único bem agrupado pelo K-Means, mas também o foi pelo BSAS, MBSAS e TTSAS usando o refinamento M ou MR. Já os conjuntos Seeds6 e Seeds8 obtiveram uma taxa menor de erros no K-Means do que no BSAS e MBSAS, respectivamente.

Tabela 6.33 VE do BSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Seeds1	**6,67**	**6,67**	9,52	9,52
Seeds2	20,48	20,48	9,52	9,52
Seeds3	48,10	39,05	88,10	40,48
Seeds4	58,57	38,57	88,10	39,05
Seeds5	20,48	20,48	10,00	10,00
Seeds6	66,19	66,19	91,90	91,90
Seeds7	21,43	21,43	13,33	13,33
Seeds8	64,76	64,76	46,19	46,19
Seeds9	64,29	64,29	46,19	46,19
Seeds10	68,10	68,10	80,00	80,00

Tabela 6.34 VE do MBSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Seeds1	21,43	21,43	14,29	14,29
Seeds2	24,29	24,29	16,19	16,19
Seeds3	74,29	74,29	71,43	71,43
Seeds4	78,10	78,10	72,86	72,86
Seeds5	18,10	18,10	12,86	12,86
Seeds6	33,81	33,81	15,24	15,24
Seeds7	12,86	12,86	**11,90**	**11,90**
Seeds8	80,00	80,00	91,90	91,90
Seeds9	85,71	85,71	92,86	92,86
Seeds10	65,71	65,71	61,43	61,43

Tabela 6.35 VE do TTSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento. M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Seeds1	31,90	13,33	58,57	13,81
Seeds2	28,57	13,33	61,90	**12,38**
Seeds3	56,19	43,81	86,67	39,05
Seeds4	72,38	40,95	79,52	38,57
Seeds5	41,43	31,90	59,52	28,57
Seeds6	80,95	80,95	90,00	86,67
Seeds7	28,57	13,33	60,95	**12,38**
Seeds8	66,19	54,29	85,71	40,48
Seeds9	66,19	54,29	85,71	39,52
Seeds10	57,14	36,67	76,19	22,38

Tabela 6.36 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	43,90	41,00	48,29	38,62
MBSAS	49,43	49,43	46,10	46,10
TTSAS	52,95	38,29	74,48	**33,38**

Tabela 6.37 VE do K-MEANS para cada um dos conjuntos de dados *Seeds*.

Conjuntos de dados	SR
Seeds1	61,43
Seeds2	61,90
Seeds3	71,90
Seeds4	72,86
Seeds5	**11,43**
Seeds6	71,90
Seeds7	91,43
Seeds8	71,90
Seeds9	90,95
Seeds10	62,38

Os índices D e DB apresentam valores bem próximos, porém é possível destacar alguns conjuntos e determinar as suas qualidades segundo esses índices. No BSAS (Tabela 6.38) o conjunto Seeds1 foi o agrupamento com melhor configuração conforme os valores de DB, enquanto que o conjunto Seeds3 e Seeds4 obtiveram altos valores de DB (baixa qualidade) quando sem uso de refinamento.

Para o MBSAS (Tabela 6.39) o conjunto mais bem estruturado segundo os índices D e DB foi o Seeds10. Os resultados do TTSAS (Tabela 6.40) comparados com os valores dos índices aplicados aos agrupamentos gerados pelo BSAS e MBSAS foram

aqueles com maior qualidade, principalmente quando avaliados pelo índice D. A Tabela 6.41 apresenta a média dos valores D e DB para os quatro esquemas.

Tabela 6.38 D e DB do BSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Seeds1	0,01170	0,59085	**0,31230**	0,59085	0,01170	**0,49497**	**0,31230**	**0,49497**
Seeds2	0,01170	1,07046	0,29620	1,07046	0,01170	0,76746	0,29620	0,76746
Seeds3	0,02542	8,86019	0,05437	1,72914	0,02542	1,50603	0,05437	1,17300
Seeds4	0,01741	4,14418	0,04481	2,09293	0,01741	1,53421	0,04481	1,06838
Seeds5	0,01149	1,09593	0,28387	1,09593	0,01149	0,73231	0,28387	0,73231
Seeds6	0,01565	2,09914	0,02985	2,09914	0,01565	0,90901	0,02985	0,90901
Seeds7	0,02043	1,21420	0,17012	1,21420	0,02043	0,78106	0,17012	0,78106
Seeds8	0,01649	2,15504	0,07888	2,15504	0,01649	0,81717	0,07888	0,81717
Seeds9	0,01635	2,13097	0,07888	2,13097	0,01635	0,84564	0,07888	0,84564
Seeds10	0,01170	1,83382	0,04736	1,83382	0,01170	1,50686	0,04736	1,50686

Tabela 6.39 D e DB do MBSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Seeds1	0,04711	0,49645	0,40263	0,49645	0,04711	0,52128	0,40263	0,52128
Seeds2	0,04652	0,48492	0,16405	0,48492	0,04652	0,52739	0,16405	0,52739
Seeds3	0,28641	0,39895	0,46402	0,39895	0,28641	0,41013	0,46402	0,41013
Seeds4	0,13513	0,43883	0,43471	0,43883	0,13513	0,45841	0,43471	0,45841
Seeds5	0,13173	0,47979	0,35751	0,47979	0,13173	0,48142	0,35751	0,48142
Seeds6	0,25359	0,78937	0,40675	0,78937	0,25359	0,68470	0,40675	0,68470
Seeds7	0,06054	0,43580	0,31230	0,43580	0,06054	0,43539	0,31230	0,43539
Seeds8	0,02269	0,62629	0,03341	0,62629	0,02269	0,58223	0,03341	0,58223
Seeds9	0,03119	0,54725	0,07277	0,54725	0,03119	0,53292	0,07277	0,53292
Seeds10	0,40718	**0,30751**	**0,47119**	**0,30751**	0,40718	0,38499	**0,47119**	0,38499

Tabela 6.40 D e DB do TTSAS para cada um dos conjuntos de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Seeds1	0,59483	0,68703	0,12179	0,35552	0,59483	0,68023	0,12179	0,34777
Seeds2	1,49934	0,70867	0,97832	0,35065	1,49934	0,67588	0,97832	0,34325
Seeds3	1,11424	0,78335	1,01734	**0,29811**	1,11424	0,69178	1,01734	0,31275
Seeds4	**1,91853**	0,61835	1,61839	0,30215	1,91853	0,66149	1,61839	0,30499
Seeds5	0,21996	0,69170	0,12179	0,35658	0,21996	0,65280	0,12179	0,38198
Seeds6	1,56077	0,64757	1,38596	0,38999	1,56077	0,57457	1,38596	0,39642
Seeds7	1,51003	0,70289	0,97874	0,35065	1,51003	0,67234	0,97874	0,34325
Seeds8	1,03762	0,76194	0,64456	0,32643	1,03762	0,63583	0,64456	0,34702
Seeds9	1,03762	0,76194	0,64456	0,32643	1,03762	0,63583	0,64456	0,35466
Seeds10	1,06807	0,78871	0,75302	0,40343	1,06807	0,67421	0,75302	0,41145

Tabela 6.41 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Seeds* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,02	2,52	0,14	0,50	**1,16**	0,72
M	0,01	1,60	0,14	0,50	0,20	**0,35**
R	0,14	0,99	0,31	0,50	0,83	0,66
MR	0,14	0,91	0,31	0,50	0,33	**0,35**

6.3.5 WDBC

Nas Tabelas 6.42 a 6.50 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 2, Θ_1: 1000 e *Close*: 500; e para o terceiro algoritmo foram: (1) Θ_1: 1500, Θ_2: 3500 e *Close*: 1500.

Os valores da VE indicam desempenhos bem distintos para os conjuntos *Wdbc* e, em alguns casos, a utilização de algum método de refinamento foi fundamental para a melhoria dos agrupamentos.

No BSAS (Tabela 6.42) pode ser observado que para o conjunto Wdbc2 as três estratégias de refinamento pioraram os seus resultados, no entanto a execução do algoritmo sem refinamento mostrou um bom comportamento. Já para os conjuntos Wdbc3, Wdbc5, Wdbc8 e Wdbc9 a utilização do refinamento R gerou uma melhoria significativa em relação à porcentagem de erros de alocação dos pontos de dados (por volta de apenas 14% de erros).

Para o MBSAS (Tabela 6.43) e TTSAS (Tabela 6.44) mais da metade dos conjuntos (seis e sete, respectivamente) não foram bem agrupados. No entanto, se no conjunto Wdbc3, por exemplo, for feita uma pequena alteração na ordem dos seus dados e utilizar como valores no TTSAS Θ_1 = 2000 e Θ_2 = 4000, os resultados (ainda que ruins) melhoram para: SR=64,1%, M=64,1%, R=66,9% e MR=66,9% (que anteriormente eram: SR=89,28%, M=89,28%, R=88,05% e MR=88,05%).

A Tabela 6.45 mostra a média dos resultados da VE para os três algoritmos com os quatro esquemas.

De maneira semelhante aos resultados obtidos pelo MBSAS e o TTSAS, aqueles obtidos pelo K-Means (Tabela 6.46) também não foram satisfatórios. Note, entretanto, que nos conjuntos Wdbc1, Wdbc7 e Wdbc9, os quatro algoritmos tiveram bons desempenhos.

Tabela 6.42 VE do BSAS para cada um dos conjuntos de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
wdbc1	36,91	36,91	19,33	19,33
wdbc2	38,14	76,80	79,09	62,74
wdbc3	37,61	76,27	14,76	62,74
wdbc4	37,96	76,80	80,49	62,74
wdbc5	37,61	76,63	16,87	62,74
wdbc6	37,43	37,43	16,52	16,52
wdbc7	36,20	36,20	15,29	15,29
wdbc8	36,91	73,99	14,76	62,74
wdbc9	34,09	67,49	**14,24**	62,74
wdbc10	36,73	36,73	20,56	20,56

Tabela 6.43 VE do MBSAS para cada um dos conjuntos de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
wdbc1	20,04	20,04	18,28	18,28
wdbc2	85,41	85,41	84,53	84,53
wdbc3	89,28	89,28	88,05	88,05
wdbc4	76,27	76,27	80,84	80,84
wdbc5	83,30	83,30	66,61	66,61
wdbc6	62,92	62,92	78,73	78,73
wdbc7	**9,49**	**9,49**	13,01	13,01
wdbc8	84,89	84,89	84,53	84,53
wdbc9	11,78	11,78	14,06	14,06
wdbc10	20,04	20,04	18,10	18,10

Tabela 6.44 VE do TTSAS para cada um dos conjuntos de dados W*dbc* considerando os quatro esquemas. SR: sem refinamento. M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
wdbc1	23,73	23,73	19,68	19,68
wdbc2	75,22	75,22	85,76	85,76
wdbc3	73,81	73,81	84,89	84,89
wdbc4	70,12	70,12	85,59	80,84
wdbc5	65,91	65,91	76,10	76,10
wdbc6	64,85	64,85	72,06	72,06
wdbc7	39,72	39,72	**17,57**	**17,57**
wdbc8	66,26	66,26	78,73	78,73
wdbc9	64,85	64,85	71,70	71,70
wdbc10	23,73	23,73	19,68	19,68

Tabela 6.45 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	36,96	59,53	**29,19**	44,82
MBSAS	54,34	54,34	54,67	54,67
TTSAS	56,82	56,82	61,18	60,70

Tabela 6.46 VE do K-MEANS para cada um dos conjuntos de dados *Wdbc*.

Conjuntos de dados	SR
wdbc1	15,99
wdbc2	84,71
wdbc3	84,36
wdbc4	84,53
wdbc5	84,53
wdbc6	84,36
wdbc7	15,64
wdbc8	84,53
wdbc9	**15,44**
wdbc10	84,36

Os valores de índices D e DB apontam que os conjuntos Wdbc1e Wdbc7 são os que apresentam melhor qualidade de agrupamento quando o algoritmo utilizado é o TTSAS (Tabela 6.49). Pela média dos resultados de D e DB (Tabela 6.50) é possível notar que quando qualquer dos três algoritmos é usado articulado ao refinamento M, os grupos do agrupamento final apresentam melhores configurações.

Tabela 6.47 D e DB do BSAS para cada um dos conjuntos de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
wdbc1	0,00544	0,22931	0,00544	0,22931	0,00512	0,33409	0,00512	0,33409
wdbc2	0,00361	1,48739	-	-	0,00148	0,59061	-	-
wdbc3	0,00447	11,71763	-	-	0,00243	0,34954	-	-
wdbc4	0,00417	3,03447	-	-	0,00131	0,59933	-	-
wdbc5	0,00509	10,45170	-	-	0,00345	0,29830	-	-
wdbc6	**0,01089**	0,38552	**0,01089**	0,38552	0,00671	**0,35311**	0,00671	**0,35311**
wdbc7	0,00656	0,63068	0,00656	0,63068	0,00783	0,36585	0,00783	0,36585
wdbc8	0,00809	1,74393	-	-	0,00776	0,31943	-	-
wdbc9	0,00329	0,91716	-	-	0,00871	0,36331	-	-
wdbc10	0,00544	0,22912	0,00544	0,22912	0,00538	0,31797	0,00538	0,31797

Tabela 6.48 D e DB do MBSAS para cada um dos conjuntos de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
wdbc1	0,01002	0,25513	0,01002	0,25513	0,00733	0,26726	0,00733	0,26726
wdbc2	0,00776	0,27557	0,00776	0,27557	0,01678	0,27366	0,01678	0,27366
wdbc3	0,00208	0,33780	0,00208	0,33780	0,00906	0,35577	0,00906	0,35577
wdbc4	0,00613	0,25115	0,00613	0,25115	0,00538	0,26661	0,00538	0,26661
wdbc5	0,00891	0,26840	0,00891	0,26840	0,01733	0,27101	0,01733	0,27101
wdbc6	**0,34767**	**0,15222**	**0,34767**	**0,15222**	0,01072	0,33861	0,01072	0,33861
wdbc7	0,00757	0,31400	0,00757	0,31400	0,01116	0,32222	0,01116	0,32222
wdbc8	0,00783	0,27568	0,00783	0,27568	0,01678	0,27519	0,01678	0,27519
wdbc9	0,00882	0,29213	0,00882	0,29213	0,01323	0,29386	0,01323	0,29386
wdbc10	0,01002	0,25513	0,01002	0,25513	0,00713	0,26832	0,00713	0,26832

Tabela 6.49 D e DB do TTSAS para cada um dos conjuntos de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
wdbc1	0,94069	0,37768	0,94069	0,37768	0,88564	0,33264	0,88564	0,33264
wdbc2	0,50168	0,27840	0,50168	0,27840	0,48753	0,33485	0,48753	0,33485
wdbc3	0,50843	0,36799	0,50843	0,36799	0,01653	0,53900	0,01653	0,53900
wdbc4	0,04529	0,53621	0,01432	0,21699	0,22969	0,37949	0,00715	**0,23394**
wdbc5	0,53566	0,37386	0,53566	0,37386	0,41338	0,36291	0,41338	0,36291
wdbc6	0,71537	0,31932	0,71537	0,31932	1,04632	0,43740	1,04632	0,43740
wdbc7	1,01425	0,38344	1,01425	0,38344	0,95841	0,29384	0,95841	0,29384
wdbc8	0,52837	0,30916	0,52837	0,30916	0,48705	0,37545	0,48705	0,37545
wdbc9	0,71537	0,31932	0,71537	0,31932	**1,20727**	0,43427	**1,20727**	0,43427
wdbc10	0,94069	0,37768	0,94069	0,37768	0,88564	0,33273	0,88564	0,33273

Tabela 6.50 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	BSAS		MBSAS		TTSAS	
de dados	D	DB	D	DB	D	DB
SR	0,00571	3,08269	**0,04168**	**0,26772**	0,64458	0,36431
M	0,00708	0,36866	**0,04168**	**0,26772**	0,64148	0,33238
R	0,00502	0,38916	0,01149	0,29325	0,66174	0,38226
MR	0,00626	0,34276	0,01149	0,29325	0,63949	0,36770

6.3.6 BREAST TUMOR

Nas Tabelas 6.51 a 6.59 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 6,

Θ_1: 5000 e *Close*: 200; e para o terceiro algoritmo foram: (1) Θ_1: 3000, Θ_2: 6000 e *Close*: 1000.

Neste conjunto de dados os valores da VE mostram que os algoritmos de agrupamentos BSAS (Tabela 6.51), MBSAS (Tabela 6.52), TTSAS (Tabela 6.53) e K-Means (Tabela 6.54) não tiveram um bom desempenho e os resultados gerados por eles foram semelhantes (apesar do BSAS apresentar um desempenho ligeiramente melhor).

Dentre os dez conjuntos de dados do *Breast*, o Breast6 foi o único que, quando usado com o BSAS sem refinamento (Tabela 6.51), produziu um agrupamento razoável. Como se trata de um conjunto com nove atributos e seis classes é difícil inferir as possíveis razões para o desempenho ruim obtido pelos quatro algoritmos. Os atributos medem características de tecido retirado da mama. Particularmente um dos atributos representa a distância (no espectro) entre valores de impedância medidos considerando as seguintes frequências: 15,625 , 31,25, 62,5, 125, 250, 500, 1000 KHz.

A Tabela 6.54 apresenta a média dos valores gerados dos conjuntos para os três algoritmos e os quatro esquemas.

Tabela 6.51 VE do BSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Breast1	67,92	67,92	73,58	73,58
Breast2	63,21	63,21	77,36	77,36
Breast3	66,98	66,98	74,53	74,53
Breast4	66,98	76,42	75,47	79,25
Breast5	78,30	79,25	75,47	74,53
Breast6	56,60	75,47	76,42	86,79
Breast7	65,09	65,09	76,42	76,42
Breast8	69,81	69,81	71,70	71,70
Breast9	**61,32**	**61,32**	70,75	70,75
Breast10	68,87	76,42	68,87	76,42

Tabela 6.52 VE do MBSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Breast1	73,58	73,58	73,58	73,58
Breast2	70,75	70,75	69,81	69,81
Breast3	75,47	75,47	74,53	74,53
Breast4	72,64	72,64	72,64	72,64
Breast5	73,58	73,58	71,70	71,70
Breast6	90,57	90,57	91,51	91,51
Breast7	79,25	79,25	86,79	86,79
Breast8	70,75	70,75	69,81	69,81
Breast9	89,62	89,62	85,85	85,85
Breast10	70,75	70,75	**68,87**	**68,87**

Tabela 6.53 VE do TTSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Breast1	**74,53**	**74,53**	76,42	76,42
Breast2	75,47	75,47	77,36	77,36
Breast3	**74,53**	**74,53**	75,47	75,47
Breast4	75,47	75,47	77,36	77,36
Breast5	75,47	75,47	76,42	76,42
Breast6	89,62	89,62	90,57	90,57
Breast7	88,68	88,68	93,40	93,40
Breast8	75,47	75,47	77,36	77,36
Breast9	86,79	86,79	89,62	89,62
Breast10	75,47	75,47	77,36	77,36

Tabela 6.54 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Wdbc* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	**66,51**	70,19	74,06	76,13
MBSAS	76,70	76,70	76,51	76,51
TTSAS	79,15	79,15	81,13	81,13

Tabela 6.55 VE do K-MEANS para cada um dos conjuntos de dados *Breast*.

Conjuntos de dados	SR
Breast1	87,74
Breast2	**71,70**
Breast3	79,25
Breast4	**71,70**
Breast5	75,47
Breast6	81,13
Breast7	73,58
Breast8	80,19
Breast9	76,42
Breast10	78,30

Os valores de índices D e DB via de regra evidenciam agrupamentos com pouca qualidade. Já pela VE ocorreram muitos erros de alocação dos pontos de dados em todos os conjuntos. É importante lembrar que, como descrito no UCI Repository, esse arquivo de dados pode ser usado para predizer a classificação em seis classes ou em quatro classes. A predição em quatro classes pode ser feita por meio da junção das instâncias que representam as classes fibro-adenoma, mastopathy e glandular em uma única (mesmo porque essas três classes não podem ser discriminadas precisamente). O fato dessas três não poderem ser discriminadas precisamente com certeza influenciou no valor da VE como medida de qualidade.

Tabela 6.56 D e DB do BSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Breast1	0,04126	4,00145	0,00274	4,00145	0,04126	1,09255	0,00274	1,09255
Breast2	0,09683	2,45709	0,00454	2,45709	0,09683	0,79690	0,00454	0,79690
Breast3	0,02178	1,16142	**0,43498**	1,16142	0,02178	**0,40914**	**0,43498**	**0,40914**
Breast4	0,02499	5,97846	0,00276	2,10014	0,02499	1,07774	0,00276	0,80913
Breast5	0,02279	9,56026	0,00235	2,44701	0,02279	0,80512	0,00235	0,75778
Breast6	0,03804	10,98058	0,01676	2,71365	0,03804	0,81444	0,01676	0,76003
Breast7	0,06091	1,90838	0,00450	1,90838	0,06091	0,84633	0,00450	0,84633
Breast8	0,07088	3,25684	0,00448	3,25684	0,07088	0,84005	0,00448	0,84005
Breast9	0,00240	2,08925	0,02132	2,08925	0,00240	0,64227	0,02132	0,64227
Breast10	0,07201	9,60719	0,07201	2,40455	0,07201	9,60719	0,07201	2,40455

Tabela 6.57 D e DB do MBSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Breast1	9,53066	0,25236	9,53066	0,25236	9,53066	0,25236	9,53066	0,25236
Breast2	0,00632	0,63484	0,31200	0,63484	0,00632	0,78368	0,31200	0,78368
Breast3	9,53066	0,25947	9,53066	0,25947	9,53066	0,26856	9,53066	0,26856
Breast4	1,39076	0,37982	1,39076	0,37982	1,39076	0,37982	1,39076	0,37982
Breast5	0,00632	0,64126	0,31200	0,64126	0,00632	0,79224	0,31200	0,79224
Breast6	20,97453	0,23204	21,11067	0,23204	20,97453	0,23163	21,11067	0,23163
Breast7	11,67435	**0,19301**	**24,16674**	**0,19301**	11,67435	0,29994	**24,16674**	0,29994
Breast8	0,00632	0,63317	0,31200	0,63317	0,00632	0,78247	0,31200	0,78247
Breast9	0,15500	0,60712	0,20869	0,60712	0,15500	0,72371	0,20869	0,72371
Breast10	0,00632	0,63533	0,31200	0,63533	0,00632	0,78445	0,31200	0,78445

Tabela 6.58 D e DB do TTSAS para cada um dos conjuntos de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	SR		M		R		MR	
de dados	D	DB	D	DB	D	DB	D	DB
Breast1	4,63010	0,28490	5,88887	0,28490	4,63010	0,30330	5,88887	0,30330
Breast2	6,15926	0,30235	8,65365	0,30235	6,15926	0,31681	8,65365	0,31681
Breast3	3,31229	0,32316	5,15499	0,32316	3,31229	0,30962	5,15499	0,30962
Breast4	5,38146	**0,26535**	6,85511	**0,26535**	5,38146	0,31141	6,85511	0,31141
Breast5	6,15926	0,31251	8,65365	0,31251	6,15926	0,31223	8,65365	0,31223
Breast6	6,54364	0,33940	6,64793	0,33940	6,54364	0,35813	6,64793	0,35813
Breast7	1,61727	0,29607	1,71597	0,29607	1,61727	0,31105	1,71597	0,31105
Breast8	6,15926	0,28367	**7,37662**	0,28367	6,15926	0,32295	**7,37662**	0,32295
Breast9	1,39057	0,30558	1,35724	0,30558	1,39057	0,31455	1,35724	0,31455
Breast10	6,15926	0,27030	7,30906	0,27030	6,15926	0,30112	7,30906	0,30112

Tabela 6.59 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Breast* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos	BSAS		MBSAS		TTSAS	
de dados	D	DB	D	DB	D	DB
SR	0,04519	5,10009	5,32812	0,44684	4,75124	**0,29833**
M	0,04775	2,45398	5,32812	0,44684	4,75124	**0,29833**
R	0,05664	1,69317	**6,71862**	0,52989	5,96131	0,31612
MR	0,05673	0,93587	**6,71862**	0,52989	5,96131	0,31612

6.4 Experimentos e Análises de Resultados dos Conjuntos Gerados Artificialmente

Esta seção apresenta os resultados obtidos e as análises dos experimentos realizados nos conjuntos de dados sintéticos descritos na Seção 6.2.

6.4.1 SINTÉTICO1A

Nas Tabelas 6.60 a 6.68 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 5, Θ_1: 1,5 e *Close*: 0,5; e para o terceiro algoritmo foram: (1) Θ_1: 1, Θ_2: 2 e *Close*: 0,5.

Com o uso do BSAS (Tabela 6.60) o melhor desempenho da VE foi obtido em apenas um conjunto de dados (dentre os dez) e, via de regra, o algoritmo teve um desempenho aquém do que poderia ser considerado razoável. Com o uso das estratégias de refinamento R e MR, entretanto, houve uma melhoria da maioria dos resultados

obtidos, o que evidencia de certa forma a importância da ordem dos dados e dos valores de parâmetros utilizados na qualidade do agrupamento obtido.

Os valores da VE mostram que o uso do MBSAS (Tabela 6.61) produziu excelentes resultados de agrupamentos. O uso das estratégias de refinamento não influenciou os resultados obtidos.

Os valores da VE mostram que o uso do TTSAS (Tabela 6.62) produziu excelentes resultados de agrupamentos. As estratégias de refinamento R e MR pioraram os resultados obtidos do TTSAS em alguns casos.

O MBSAS e o TTSAS apresentaram comportamento similares nos mesmos conjuntos (erros entre 0 e 7%).

Apenas para enfatizar a importância de valor de parâmetros, considere a seguinte situação: arquivo de dados Sintético1a_5, algoritmo de agrupamento BSAS, Θ_1 = 1,5 e resultado obtido: SR=98%, M=98%, R=86% e MR=86%. Para Θ_1 = 4, por exemplo, o seguinte resultado é obtido: SR=43,73%, M=40,73%, R=27,67% e MR=27,67%.

A Tabela 6.57 apresenta a média da VE para os conjuntos de dados nos quatro esquemas, sendo possível verificar que para o Sintético1a o MBSAS obteve os melhores resultados e em pelo menos cinco dos conjuntos de dados a sensibilidade na ordem de apresentação dos pontos de dados foi bem contornada. Em comparação com o K-Means (Tabela 6.64), o MBSAS e o TTSAS foram mais eficientes como aponta os resultados da VE.

Tabela 6.60 VE do BSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1a_1	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_2	91,33	91,33	59,67	59,67
Sintético1a_3	93,67	92,00	42,33	25,67
Sintético1a_4	91,67	91,67	20,33	20,33
Sintético1a_5	98,00	98,00	86,00	86,00
Sintético1a_6	90,33	78,67	55,67	38,67
Sintético1a_7	72,00	72,00	59,67	59,67
Sintético1a_8	63,00	63,00	59,67	59,67
Sintético1a_9	96,00	96,00	79,67	79,67
Sintético1a_10	97,33	95,00	83,67	80,00

Tabela 6.61 VE do MBSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1a_1	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_2	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_3	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_4	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_5	40,00	40,00	40,00	40,00
Sintético1a_6	38,67	38,67	38,67	38,67
Sintético1a_7	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_8	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1a_9	60,00	60,00	60,00	60,00
Sintético1a_10	60,00	60,00	60,00	60,00

Tabela 6.62 VE do TTSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Esquema de resultado	SR	M	R	MR
Sintético1a_1	6,67	6,67	39,67	39,67
Sintético1a_2	4,33	4,33	37,33	37,33
Sintético1a_3	4,00	4,00	24,00	24,00
Sintético1a_4	4,00	4,00	86,67	86,67
Sintético1a_5	46,00	46,00	66,67	66,67
Sintético1a_6	41,33	41,33	51,00	51,00
Sintético1a_7	6,33	6,33	36,33	36,33
Sintético1a_8	**3,00**	**3,00**	25,33	25,33
Sintético1a_9	60,67	60,67	73,00	73,00
Sintético1a_10	61,33	61,33	82,00	82,00

Tabela 6.63 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	79,33	77,77	54,67	50,93
MBSAS	**19,87**	**19,87**	**19,87**	**19,87**
TTSAS	23,77	23,77	52,20	52,20

Tabela 6.64 VE do K-MEANS para cada um dos conjuntos de dados *Sintético1a*.

Conjuntos de dados	SR
Sintético1a_1	80,33
Sintético1a_2	68,67
Sintético1a_3	68,33
Sintético1a_4	60,00
Sintético1a_5	**52,67**
Sintético1a_6	72,00
Sintético1a_7	80,00
Sintético1a_8	80,00
Sintético1a_9	60,00
Sintético1a_10	71,67

Os índices D e DB indicam que bons agrupamentos foram obtidos em vários conjuntos nos três algoritmos, principalmente no MBSAS e TTSAS. Particularmente o BSAS (Tabela 6.65) no conjunto Sintético1a_1, o MBSAS (Tabela 6.66) e TTSAS (Tabela 6.67) em todos os conjuntos.

Tabela 6.65 D e DB do BSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1a_1	**2,61441**	**0,15867**	**2,61441**	**0,15867**	**2,61441**	**0,15867**	**2,61441**	**0,15867**
Sintético1a_2	0,34528	0,94513	0,34528	0,94513	0,57976	0,38562	0,57976	0,38562
Sintético1a_3	0,00240	32,29177	0,00087	1,62211	0,65972	0,69053	0,01457	0,44108
Sintético1a_4	0,00176	0,93025	0,00176	0,93025	0,54252	0,27752	0,54252	0,27752
Sintético1a_5	0,24519	1,20906	0,24519	1,20906	0,77981	0,45167	0,77981	0,45167
Sintético1a_6	0,00068	1,08068	0,00068	1,08068	0,31323	0,27597	0,71332	0,40542
Sintético1a_7	0,00307	1,00117	0,00307	1,00117	0,15180	0,41703	0,15180	0,41703
Sintético1a_8	0,66576	0,56998	0,66576	0,56998	0,77981	0,32264	0,77981	0,32264
Sintético1a_9	0,00286	1,14736	0,00286	1,14736	0,64197	1,14736	0,64197	1,14736
Sintético1a_10	0,00521	3,48225	0,00521	0,75894	0,49547	0,48090	0,49547	0,45924

Tabela 6.66 D e DB do MBSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1a_1	2,61441	0,15867	2,61441	0,15867	2,61441	0,15867	2,61441	0,15867
Sintético1a_2	2,61441	0,15868	2,61441	0,15868	2,61441	0,15868	2,61441	0,15868
Sintético1a_3	2,61441	0,15907	2,61441	0,15907	2,61441	0,15907	2,61441	0,15907
Sintético1a_4	2,61441	0,15928	2,61441	0,15928	2,61441	0,15928	2,61441	0,15928
Sintético1a_5	2,61441	**0,14280**	2,61441	**0,14280**	2,61441	**0,14280**	2,61441	**0,14280**
Sintético1a_6	2,61441	0,15817	2,61441	0,15817	2,61441	0,15817	2,61441	0,15817
Sintético1a_7	2,61441	0,15901	2,61441	0,15901	2,61441	0,15901	2,61441	0,15901
Sintético1a_8	2,61441	0,15943	2,61441	0,15943	2,61441	0,15943	2,61441	0,15943
Sintético1a_9	**3,91054**	0,14518	**3,91054**	0,14518	**3,91054**	0,14518	**3,91054**	0,14518
Sintético1a_10	**3,91054**	0,14472	**3,91054**	0,14472	**3,91054**	0,14472	**3,91054**	0,14472

Tabela 6.67 D e DB do TTSAS para cada um dos conjuntos de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1a_1	3,39198	0,37689	3,39198	0,37689	2,74121	0,40910	2,74121	0,40910
Sintético1a_2	2,43876	0,36486	2,43876	0,36486	2,47276	0,46426	2,47276	0,46426
Sintético1a_3	2,01952	0,36134	2,01952	0,36134	2,00101	0,39338	2,00101	0,39338
Sintético1a_4	2,97767	**0,28060**	2,97767	**0,28060**	2,54717	0,37509	2,54717	0,37509
Sintético1a_5	2,01952	0,38831	2,01952	0,38831	2,00101	0,41777	2,00101	0,41777
Sintético1a_6	3,42583	0,28117	3,42583	0,28117	2,55049	0,36595	2,55049	0,36595
Sintético1a_7	3,04179	0,36758	3,04179	0,36758	2,74121	0,42294	2,74121	0,42294
Sintético1a_8	3,39198	0,32691	3,39198	0,32691	3,00024	0,37165	3,00024	0,37165
Sintético1a_9	3,39198	0,35230	3,39198	0,35230	2,74121	0,41514	2,74121	0,41514
Sintético1a_10	**3,59727**	0,41930	**3,59727**	0,41930	2,97674	0,47024	2,97674	0,47024

Tabela 6.68 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético1a* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,44310	1,17331	**3,02528**	**0,15355**	2,73318	0,36920
M	0,44310	0,87072	**3,02528**	**0,15355**	2,73318	0,36920
R	0,76371	0,46985	**3,02528**	**0,15355**	2,46774	0,40477
MR	0,76371	0,46744	**3,02528**	**0,15355**	2,46774	0,40477

6.4.2 SINTÉTICO1B

Nas tabelas 6.69 a 6.93 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 5, Θ_1: 1,5 e *Close*: 0,5; e para o terceiro algoritmo foram: (1) Θ_1: 1, Θ_2: 2 e *Close*: 0,5. Para cada algoritmo são apresentados os resultados dos quatro esquemas em três diferentes ambientes, a saber: (1) sem pré-processamento (SPP), (2) com remoção de dados (CR) e (3) com substituição do valor ausente do atributo (CS).

Como mostrado na Figura 6.1(b) o Sintético1b é formado pela adição de ruídos (i.e. atributos com valores ausentes) em 10% dos pontos de dados que formam o conjunto Sintético1a. Pelos valores da VE, no BSAS sem pré-processamento de dados (Tabela 6.69) os conjuntos de dados não foram bem agrupamentos (exceto o Sintéticob_3 com refinamento R e MR).

Comparando os valores para o Sintético1a_1 no BSAS (Tabela 6.60), que não possui valores de atributos ausentes, e para o Sintético1b_1, que possui valores de atributos ausentes e gerado sem pré-processamento de dados, nota-se que esse problema realmente pode influenciar nos resultados. No entanto, se for utilizado um pré-processamento com a remoção dos dados com valores ausentes (Tabela 6.70), o Sintético1b_1 fica agrupado com 0% de erros de alocação. Assim, no caso do conjunto de dados Sintético1b, para minimizar o efeito adverso dos valores ausentes, na prática, é sempre bom a remoção dos pontos de dados antes do processo de agrupamento realizado pelo BSAS. Situação semelhante ocorre com o MBSAS (Tabela 6.73) e o TTSAS (Tabela 6.76).

As tabelas 6.78, 6.79 e 6.80 mostram as médias da VE para os três algoritmos nos quatro esquemas, em situações: sem pré-processamento de dados (SPP), com remoção de dado (CR) e com a substituição do valor ausente do atributo (CS).

O benefício que a remoção de dados com atributos ausentes pode trazer pode também ser evidenciado na Tabela 6.81, que mostra os resultados do K-Means considerando os três diferentes ambientes, i.e., SPP, CR e CS.

Tabela 6.69 VE do BSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	77,00	76,33	79,67	79,67
Sintético1b_2	90,67	90,67	65,33	65,33
Sintético1b_3	93,33	93,33	**33,33**	**33,33**
Sintético1b_4	95,67	95,67	61,00	61,00
Sintético1b_5	96,33	88,33	38,67	66,00
Sintético1b_6	95,00	95,00	59,00	59,00
Sintético1b_7	79,00	71,67	78,67	78,33
Sintético1b_8	78,67	78,67	79,67	79,67
Sintético1b_9	95,67	80,33	78,33	96,33
Sintético1b_10	96,33	96,33	82,67	82,67

Tabela 6.70 VE do BSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_2	91,48	91,48	60,74	60,74
Sintético1b_3	93,70	91,85	42,59	25,19
Sintético1b_4	92,59	92,59	20,37	20,37
Sintético1b_5	98,15	98,15	85,93	85,93
Sintético1b_6	90,00	77,41	54,81	38,52
Sintético1b_7	72,22	72,22	59,63	59,63
Sintético1b_8	64,44	64,44	60,00	60,00
Sintético1b_9	96,30	96,30	79,63	79,63
Sintético1b_10	97,04	95,19	83,33	80,00

Tabela 6.71 VE do BSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	99,00	80,67	82,33	80,00
Sintético1b_2	99,67	98,67	80,00	98,67
Sintético1b_3	99,67	99,67	80,00	80,00
Sintético1b_4	95,67	95,67	69,67	69,67
Sintético1b_5	96,33	90,67	72,00	82,33
Sintético1b_6	89,00	89,00	**23,00**	**23,00**
Sintético1b_7	99,00	80,67	74,67	62,00
Sintético1b_8	96,67	80,67	53,00	80,00
Sintético1b_9	99,33	99,33	79,67	79,67
Sintético1b_10	100,00	100,00	81,67	81,67

Tabela 6.72 VE do MBSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	42,67	42,67	42,67	42,67
Sintético1b_2	75,00	75,00	63,00	63,00
Sintético1b_3	89,67	89,67	91,00	91,00
Sintético1b_4	**24,00**	**24,00**	**24,00**	**24,00**
Sintético1b_5	77,67	77,67	77,67	77,67
Sintético1b_6	59,00	59,00	59,00	59,00
Sintético1b_7	42,67	42,67	42,67	42,67
Sintético1b_8	78,33	78,33	79,67	79,67
Sintético1b_9	97,33	97,33	97,33	97,33
Sintético1b_10	97,33	97,33	97,33	97,33

Tabela 6.73 VE do MBSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_2	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_3	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_4	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_5	40,00	40,00	40,00	40,00
Sintético1b_6	38,52	38,52	38,52	38,52
Sintético1b_7	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_8	**0,00**	**0,00**	**0,00**	**0,00**
Sintético1b_9	60,00	60,00	60,00	60,00
Sintético1b_10	60,00	60,00	60,00	60,00

Tabela 6.74 VE do MBSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	62,00	62,00	60,67	60,67
Sintético1b_2	62,00	80,33	67,67	79,67
Sintético1b_3	**43,33**	62,67	47,00	62,00
Sintético1b_4	62,00	62,00	60,67	60,67
Sintético1b_5	62,00	62,00	63,33	63,33
Sintético1b_6	54,67	54,67	56,67	56,67
Sintético1b_7	62,00	62,00	60,67	60,67
Sintético1b_8	62,00	62,00	60,67	60,67
Sintético1b_9	97,67	97,67	100,00	100,00
Sintético1b_10	96,67	96,67	96,33	96,33

Tabela 6.75 VE do TTSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	48,67	48,67	58,67	58,67
Sintético1b_2	99,00	99,00	99,00	99,00
Sintético1b_3	63,00	63,00	63,00	63,00
Sintético1b_4	65,00	65,00	75,67	75,67
Sintético1b_5	83,67	83,67	93,67	93,67
Sintético1b_6	60,67	60,67	61,00	61,00
Sintético1b_7	48,67	48,67	58,67	58,67
Sintético1b_8	**46,00**	**46,00**	**46,00**	**46,00**
Sintético1b_9	97,33	97,33	97,33	97,33
Sintético1b_10	80,67	80,67	89,00	89,00

Tabela 6.76 VE do TTSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	6,30	6,30	39,63	39,63
Sintético1b_2	2,96	2,96	25,19	25,19
Sintético1b_3	**2,22**	**2,22**	12,22	12,22
Sintético1b_4	4,44	4,44	29,26	29,26
Sintético1b_5	45,56	45,56	66,30	66,30
Sintético1b_6	38,52	38,52	38,52	38,52
Sintético1b_7	5,93	5,93	35,19	35,19
Sintético1b_8	2,96	2,96	24,81	24,81
Sintético1b_9	60,74	60,74	73,33	73,33
Sintético1b_10	61,48	61,48	82,59	82,59

Tabela 6.77 VE do TTSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético1b_1	84,33	84,33	94,00	94,00
Sintético1b_2	98,33	98,33	98,67	98,67
Sintético1b_3	97,00	97,00	97,33	97,33
Sintético1b_4	84,33	84,33	94,33	94,33
Sintético1b_5	83,67	83,67	93,67	93,67
Sintético1b_6	56,33	**56,33**	61,00	61,00
Sintético1b_7	84,33	84,33	94,00	94,00
Sintético1b_8	81,33	81,33	81,33	81,33
Sintético1b_9	98,00	98,00	98,00	98,00
Sintético1b_10	98,00	98,00	98,00	98,00

Tabela 6.78 Média de erro VE do BSAS, MBSAS e TTSAS (SPP) para o conjunto de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	89,77	86,63	**65,63**	70,13
MBSAS	68,37	68,37	67,43	67,43
TTSAS	69,27	69,27	74,20	74,20

Tabela 6.79 Média de erro VE do BSAS, MBSAS e TTSAS (CR) para o conjunto de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Método	SR	M	R	MR
BSAS	79,59	77,96	54,70	51,00
MBSAS	**19,85**	**19,85**	**19,85**	**19,85**
TTSAS	23,11	23,11	42,70	42,70

Tabela 6.80 Média de erro VE do BSAS, MBSAS e TTSAS (CS) para o conjunto de dados *Sintético1b*. considerando os quatro esquemas SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	97,43	91,50	69,60	73,70
MBSAS	**66,43**	70,20	67,37	70,07
TTSAS	86,57	86,57	91,03	91,03

Tabela 6.81 VE do K-MEANS para cada um dos conjuntos de dados *Sintético1b*. SPP: sem pré-processamento de dados, CR: com remoção do dado e CS: com Substituição do valor ausente do atributo.

Conjuntos de dados	SPP	CR	CS
Sintético1b_1	62,33	80,00	78,00
Sintético1b_2	64,33	70,37	80,00
Sintético1b_3	82,00	77,78	81,00
Sintético1b_4	79,33	72,96	**24,00**
Sintético1b_5	97,00	32,59	44,00
Sintético1b_6	80,33	74,07	80,00
Sintético1b_7	64,33	69,26	80,67
Sintético1b_8	97,00	80,00	80,33
Sintético1b_9	78,67	60,74	79,33
Sintético1b_10	81,67	48,52	80,33

Quando valores ausentes são produzidos, a separação intergrupos pode diminuir muito e a dispersão de um grupo (ou dos grupos) aumentar. Isso se deve ao índice D utilizar apenas a distância mínima de pares (ao invés da distância média de pares entre os pontos de dados em diferentes grupos) e o índice DB ser baseado em uma medida de similaridade entre grupos que, por sua vez, se baseia na medida de dispersão de um grupo (o DB mede a similaridade média entre cada grupo e aquele que lhe é mais semelhante).

Assim, como os grupos supostamente devem ser compactos e bem separados, quanto maior for a incidência de valores ausentes nos atributos, maior será o impacto negativo sofrido pelas estruturas dos grupos. As tabelas 6.91, 6.92 e 6.93 mostram as médias de D e DB para os três algoritmos nos quatro esquemas, considerando os três ambientes: SPP, CR e CS, respectivamente.

Tabela 6.82 D e DB do BSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de	SR		M		R		MR	
dados	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	0,00397	1,66552	0,00841	1,00547	0,00397	1,00547	0,00841	0,41034
Sintético1b_2	0,02042	2,30548	0,33430	2,30548	0,02042	2,30548	0,33430	0,69878
Sintético1b_3	0,00211	2,21314	0,30215	2,21314	0,00211	2,21314	0,30215	0,69514
Sintético1b_4	0,00910	0,87656	0,88517	0,87656	0,00910	0,87656	0,88517	0,34886
Sintético1b_5	0,00304	1,30595	0,45513	1,18451	0,00304	1,18451	0,45513	0,45264
Sintético1b_6	0,00382	0,88123	**0,73314**	0,88123	0,00382	0,88123	**0,73314**	0,45929
Sintético1b_7	0,00780	19,37545	0,13959	2,00014	0,00780	2,00014	0,13959	0,43650
Sintético1b_8	0,00397	0,00397	0,02476	**0,00397**	0,00397	**0,00397**	0,02476	0,45960
Sintético1b_9	0,00374	1,94268	0,48151	1,34500	0,00374	1,34500	0,48151	0,38388
Sintético1b_10	0,00213	0,98315	0,63468	0,98315	0,00213	0,98315	0,63468	0,71220

Tabela 6.83 D e DB do BSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	**2,61441**	**0,15821**	**2,61441**	**0,15821**	**2,61441**	**0,15821**	**2,61441**	**0,15821**
Sintético1b_2	0,34528	0,43548	0,58045	0,43548	0,34528	0,43548	0,58045	0,43548
Sintético1b_3	0,00240	5,50695	0,73230	1,63370	0,00240	0,72549	0,73230	0,43773
Sintético1b_4	0,00176	0,94027	0,55525	0,94027	0,00176	0,27578	0,55525	0,27578
Sintético1b_5	0,27080	1,19011	0,77981	1,19011	0,27080	0,43376	0,77981	0,43376
Sintético1b_6	0,00068	2,56095	0,33697	1,05091	0,00068	0,28275	0,33697	0,39924
Sintético1b_7	0,00457	1,03093	0,20085	1,03093	0,00457	0,40388	0,20085	0,40388
Sintético1b_8	0,66576	0,55502	0,77981	0,55502	0,66576	0,35762	0,77981	0,35762
Sintético1b_9	0,00286	1,15280	0,64197	1,15280	0,00286	0,43734	0,64197	0,43734
Sintético1b_10	0,00525	3,56636	0,49547	0,75067	0,00525	0,46759	0,49547	0,43847

Tabela 6.84 D e DB do BSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	0,01371	5,15050	0,01932	-	0,01371	0,60805	0,01932	-
Sintético1b_2	0,00434	2,44146	0,27026	1,00794	0,00434	0,40838	0,27026	0,37233
Sintético1b_3	0,00776	1,36579	0,04125	1,36579	0,00776	0,36261	0,04125	**0,36261**
Sintético1b_4	0,00910	0,87656	0,43429	0,87656	0,00910	0,50624	0,43429	0,50624
Sintético1b_5	0,00117	1,25632	0,35632	1,00990	0,00117	0,55572	0,35632	0,49191
Sintético1b_6	0,00939	1,10554	0,56101	1,10554	0,00939	0,40815	**0,56101**	0,40815
Sintético1b_7	0,02537	2,40049	0,32531	0,54970	0,02537	0,42298	0,32531	0,46266
Sintético1b_8	0,01390	9,84885	0,01944	-	0,01390	0,64423	0,01944	-
Sintético1b_9	0,00416	0,87082	0,00145	0,87082	0,00416	0,66682	0,00145	0,66682
Sintético1b_10	0,00076	1,12079	0,38725	1,12079	0,00076	0,39457	0,38725	0,39457

Tabela 6.85 D e DB do MBSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	1,35811	0,27072	1,35811	0,27072	1,35811	0,27072	1,35811	0,27072
Sintético1b_2	0,00304	0,45832	0,64197	0,45832	0,00304	0,43383	0,64197	0,43383
Sintético1b_3	0,77981	0,50168	0,77981	0,50168	0,77981	0,51617	0,77981	0,51617
Sintético1b_4	0,77981	0,31724	0,77981	0,31724	0,77981	0,31724	0,77981	0,31724
Sintético1b_5	**1,67735**	**0,22895**	**1,67735**	**0,22895**	**1,67735**	**0,22895**	**1,67735**	**0,22895**
Sintético1b_6	0,73314	0,31648	0,73314	0,31648	0,73314	0,31648	0,73314	0,31648
Sintético1b_7	1,35811	0,27072	1,35811	0,27072	1,35811	0,27072	1,35811	0,27072
Sintético1b_8	0,47766	0,35033	0,73310	0,35033	0,47766	0,32750	0,73310	0,32750
Sintético1b_9	0,64197	0,39882	0,64197	0,39882	0,64197	0,39882	0,64197	0,39882
Sintético1b_10	0,64197	0,39995	0,64197	0,39995	0,64197	0,39995	0,64197	0,39995

Tabela 6.86 D e DB do MBSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	2,61441	0,15821	2,61441	0,15821	2,61441	0,15821	2,61441	0,15821
Sintético1b_2	2,61441	0,15792	2,61441	0,15792	2,61441	0,15792	2,61441	0,15792
Sintético1b_3	2,61441	0,15867	2,61441	0,15867	2,61441	0,15867	2,61441	0,15867
Sintético1b_4	2,61441	0,15847	2,61441	0,15847	2,61441	0,15847	2,61441	0,15847
Sintético1b_5	2,61441	**0,14263**	2,61441	**0,14263**	2,61441	**0,14263**	2,61441	**0,14263**
Sintético1b_6	2,61441	0,15820	2,61441	0,15820	2,61441	0,15820	2,61441	0,15820
Sintético1b_7	2,61441	0,15861	2,61441	0,15861	2,61441	0,15861	2,61441	0,15861
Sintético1b_8	2,61441	0,15902	2,61441	0,15902	2,61441	0,15902	2,61441	0,15902
Sintético1b_9	**3,91054**	0,14478	**3,91054**	0,14478	**3,91054**	0,14478	**3,91054**	0,14478
Sintético1b_10	**3,91054**	0,14440	**3,91054**	0,14440	**3,91054**	0,14440	**3,91054**	0,14440

Tabela 6.87 D e DB do MBSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	0,63602	0,34028	0,63602	0,34028	0,63602	0,39559	0,63602	0,39559
Sintético1b_2	0,40315	0,41686	0,63602	0,41686	0,40315	0,51110	0,63602	0,51110
Sintético1b_3	0,03297	0,41617	0,60710	0,41617	0,03297	0,48644	0,60710	0,48644
Sintético1b_4	0,63602	0,34029	0,63602	0,34029	0,63602	0,39559	0,63602	0,39559
Sintético1b_5	0,85339	**0,29339**	0,65556	**0,29339**	0,85339	0,38159	0,65556	0,38159
Sintético1b_6	0,00313	0,48765	0,61911	0,48765	0,00313	0,52797	0,61911	0,52797
Sintético1b_7	0,63602	0,34028	0,63602	0,34028	0,63602	0,39559	0,63602	0,39559
Sintético1b_8	0,63602	0,34070	0,63602	0,34070	0,63602	0,39626	0,63602	0,39626
Sintético1b_9	0,38808	0,46725	**1,55251**	0,46725	0,38808	0,41539	**1,55251**	0,41539
Sintético1b_10	0,63535	0,34170	0,63535	0,34170	0,63535	0,36808	0,63535	0,36808

Tabela 6.88 D e DB do TTSAS (SPP) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	3,54156	0,28637	4,13496	0,28637	3,54156	0,31890	4,13496	0,31890
Sintético1b_2	3,76056	0,35883	2,89629	0,35883	3,76056	0,38149	2,89629	0,38149
Sintético1b_3	1,05078	0,38059	1,05078	0,38059	1,05078	0,38059	1,05078	0,38059
Sintético1b_4	3,44561	0,30884	3,80106	0,30884	3,44561	0,34380	3,80106	0,34380
Sintético1b_5	3,19349	0,29755	3,71905	0,29755	3,19349	0,32819	3,71905	0,32819
Sintético1b_6	0,55730	0,28488	0,64161	0,28488	0,55730	0,32872	0,64161	0,32872
Sintético1b_7	3,18593	0,27924	4,13496	0,27924	3,18593	0,32797	4,13496	0,32797
Sintético1b_8	**4,10392**	**0,27803**	**4,38683**	**0,27803**	4,10392	0,31015	**4,38683**	0,31015
Sintético1b_9	3,76625	0,36929	2,81156	0,36929	3,76625	0,38731	2,81156	0,38731
Sintético1b_10	1,10006	0,41789	0,97437	0,41789	1,10006	0,43153	0,97437	0,43153

Tabela 6.89 D e DB do TTSAS (CR) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	3,39198	0,37167	2,74121	0,37167	3,39198	0,41102	2,74121	0,41102
Sintético1b_2	3,09063	0,35415	2,59269	0,35415	3,09063	0,40011	2,59269	0,40011
Sintético1b_3	2,77163	0,32252	2,61441	0,32252	2,77163	0,31615	2,61441	0,31615
Sintético1b_4	3,10179	0,32261	2,58069	0,32261	3,10179	0,37421	2,58069	0,37421
Sintético1b_5	2,51543	0,38600	2,43876	0,38600	2,51543	0,42081	2,43876	0,42081
Sintético1b_6	0,09439	0,27885	0,02343	0,27885	0,09439	0,32633	0,02343	0,32633
Sintético1b_7	3,04179	0,36119	2,74121	0,36119	3,04179	0,42766	2,74121	0,42766
Sintético1b_8	3,39198	0,32300	3,00024	0,32300	3,39198	0,37291	3,00024	0,37291
Sintético1b_9	3,39198	**0,30772**	2,74121	**0,30772**	3,39198	0,34817	2,74121	0,34817
Sintético1b_10	**3,59727**	0,39004	2,90711	0,39004	**3,59727**	0,42058	2,90711	0,42058

Tabela 6.90 D e DB do TTSAS (CS) para cada um dos conjuntos de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético1b_1	0,89847	0,39366	0,50687	0,39366	0,89847	0,39753	0,50687	0,39753
Sintético1b_2	1,18511	0,44498	1,53867	0,44498	1,18511	0,41021	1,53867	0,41021
Sintético1b_3	1,01297	0,45952	0,96477	0,45952	1,01297	0,41684	0,96477	0,41684
Sintético1b_4	2,86443	0,42749	2,99541	0,42749	2,86443	0,39150	2,99541	0,39150
Sintético1b_5	1,02440	0,47985	**3,71905**	0,47985	1,02440	0,32819	**3,71905**	0,32819
Sintético1b_6	2,47540	0,39823	1,99336	0,39823	2,47540	0,40002	1,99336	0,40002
Sintético1b_7	0,84836	0,44825	1,31001	0,44825	0,84836	0,40721	1,31001	0,40721
Sintético1b_8	1,19111	0,40674	1,50872	0,40674	1,19111	0,39989	1,50872	0,39989
Sintético1b_9	0,89847	**0,32585**	0,37175	**0,32585**	0,89847	0,38111	0,37175	0,38111
Sintético1b_10	1,02061	0,38863	1,56203	0,38863	1,02061	0,40310	1,56203	0,40310

Tabela 6.91 Média do D e DB do BSAS, MBSAS e TTSAS (SPP) para o conjunto de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,00601	3,15531	0,84510	0,35132	2,77055	**0,32615**
M	0,01969	1,27987	0,84510	0,35132	2,77055	**0,32615**
R	0,39988	1,27987	0,93453	0,34804	**2,85515**	0,35386
MR	0,38755	0,50572	0,93453	0,34804	**2,85515**	0,35386

Tabela 6.92 Média do D e DB do BSAS, MBSAS e TTSAS (CR) para o conjunto de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,39138	1,70971	**2,87364**	**0,15409**	2,83889	0,34177
M	0,39122	0,88981	**2,87364**	**0,15409**	2,83889	0,34177
R	0,77173	0,39779	**2,87364**	**0,15409**	2,43810	0,38179
MR	0,74233	0,37775	**2,87364**	**0,15409**	2,43810	0,38179

Tabela 6.93 Média do D e DB do BSAS, MBSAS e TTSAS (CS) para o conjunto de dados *Sintético1b* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,00897	2,64371	0,48601	0,37846	1,34193	0,41732
M	0,00688	0,98838	0,48601	**0,36389**	1,34193	0,41732
R	0,24159	0,49778	0,72497	0,42736	**1,64707**	0,39356
MR	0,18515	0,45816	0,72497	0,39660	**1,64707**	0,39356

6.4.3 SINTÉTICO2

Nas Tabela 6.94 a 6.103 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 3, Θ_1: 1 e *Close*: 0,5; e para o terceiro algoritmo foram: (1) Θ_1: 1, Θ_2: 2 e *Close*: 0,5.

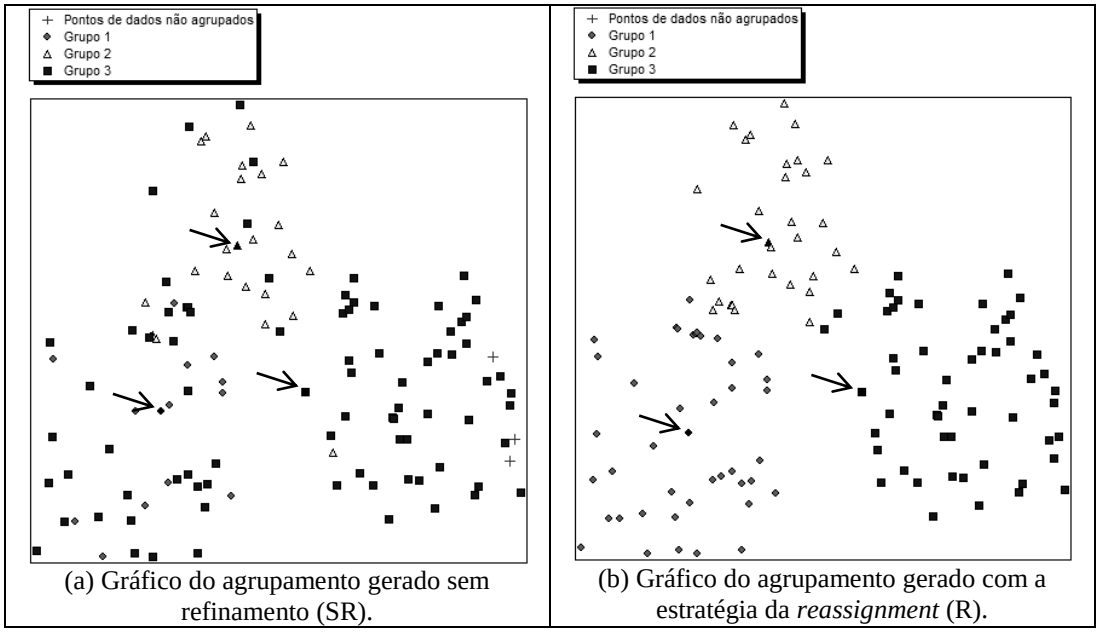
Os valores da VE mostram que, em geral, os algoritmos tiveram um excelente desempenho na alocação dos pontos de dados.

Com o uso do BSAS (Tabela 6.94) nos conjuntos Sintético2_8 e Sintético2_9 aconteceram inúmeros erros de alocação. As estratégias de refinamento, entretanto, foram fundamentais para uma boa melhoria nos resultados.

A Figura 6.5 apresenta dois gráficos de dois agrupamentos do conjunto de dados Sintético2_5 gerados usando o BSAS. O primeiro mostra o resultado do agrupamento obtido sem opção de refinamento (SR) e com VE de 32% (Figura 6.5(a)) e o segundo, obtido com a estratégia de refinamento R e com VE de 6,4% (Figura 6.5(b)). Acima de cada gráfico está a legenda de símbolos que representam cada grupo (um desses símbolos representam os dados não alocados a grupos). As três setas apontam para os centroides dos três grupos criados.

Tabela 6.94 VE do BSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético2_1	**2,4**	**2,4**	4,8	4,8
Sintético2_2	12	12	4,8	4,8
Sintético2_3	21,6	21,6	8	8
Sintético2_4	29,6	29,6	6,4	6,4
Sintético2_5	32	32	6,4	6,4
Sintético2_6	8	8	4,8	4,8
Sintético2_7	7,2	7,2	4,8	4,8
Sintético2_8	63,2	47,2	36,8	49,6
Sintético2_9	60	60	31,2	31,2
Sintético2_10	8,8	8,8	4,8	4,8



(a) Gráfico do agrupamento gerado sem refinamento (SR).

(b) Gráfico do agrupamento gerado com a estratégia da *reassignment* (R).

Figura 6.5. Gráficos dos agrupamentos do conjunto Sintético2_5 gerados no BSAS sem e com refinamento.

O pior desempenho do MBSAS (Tabela 6.95) foi com o Sintético2_7 (53,6%). O uso do TTSAS (Tabela 6.96) com os conjuntos Sintético2_2 e Sintético2_9, não produziu agrupamentos significativos.

Por outro lado o K-Means (Tabela 6.98) obteve bons resultados em apenas três conjuntos. Isso de certa forma faz com que a escolha de um algoritmo de agrupamentos para um conjunto em que grupos não estão bem definidos, deva ser entre os três algoritmos. A Tabela 6.97 mostra a média dos resultados para os quatro esquemas obtidos com o BSAS, MBSAS e TTSAS.

Tabela 6.95 VE do MBSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético2_1	16,8	16,8	10,4	10,4
Sintético2_2	12	12	8,8	8,8
Sintético2_3	**4,8**	**4,8**	**4,8**	**4,8**
Sintético2_4	8,8	8,8	8	8
Sintético2_5	4,8	4,8	**4,8**	**4,8**
Sintético2_6	17,6	17,6	10,4	10,4
Sintético2_7	48	48	50,4	53,6
Sintético2_8	9,6	9,6	8	8
Sintético2_9	16,8	16,8	10,4	10,4
Sintético2_10	26,4	26,4	14,4	14,4

Tabela 6.96 VE do TTSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético2_1	19,2	**15,2**	57,6	16,8
Sintético2_2	67,2	67,2	68,8	68,8
Sintético2_3	37,6	37,6	28	28
Sintético2_4	38,4	38,4	26,4	26,4
Sintético2_5	38,4	38,4	28,8	28,8
Sintético2_6	19,2	**15,2**	56,8	16,8
Sintético2_7	19,2	**15,2**	57,6	16,8
Sintético2_8	19,2	**15,2**	44,8	16
Sintético2_9	75,2	75,2	88,8	88,8
Sintético2_10	19,2	15,2	57,6	16,8

Tabela 6.97 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	24,48	22,88	**11,28**	12,56
MBSAS	16,56	16,56	13,04	13,36
TTSAS	35,28	33,28	51,52	32,40

Tabela 6.98 VE do K-MEANS para cada um dos conjuntos de dados *Sintético2*.

Conjuntos de dados	SR
Sintético2_1	9,60
Sintético2_2	8,80
Sintético2_3	**6,40**
Sintético2_4	56,00
Sintético2_5	76,00
Sintético2_6	70,40
Sintético2_7	71,20
Sintético2_8	74,40
Sintético2_9	71,20
Sintético2_10	55,20

Os valores dos índices D e DB evidenciam que os conjuntos Sintético2_3 e Sintético2_4 (usando o TTSAS) (Tabela 6.101) foram os que apresentaram agrupamentos com melhor estrutura em qualquer dos esquemas.

Tabela 6.99 D e DB do BSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético2_1	0,03830	0,49550	0,03830	0,49550	**0,18950**	0,48959	**0,18950**	0,48959
Sintético2_2	0,02209	0,65256	0,02209	0,65256	**0,18950**	0,55721	**0,18950**	0,55721
Sintético2_3	0,00192	0,84989	0,00192	0,84989	0,10907	0,68151	0,10907	0,68151
Sintético2_4	0,00748	0,97110	0,00748	0,97110	0,06060	0,78866	0,06060	0,78866
Sintético2_5	0,00748	1,03221	0,00748	1,03221	0,06060	0,76100	0,06060	0,76100
Sintético2_6	0,00937	0,58545	0,00937	0,58545	**0,18950**	0,53341	**0,18950**	0,53341
Sintético2_7	0,01282	0,61157	0,01282	0,61157	0,16976	0,54043	0,18950	0,53341
Sintético2_8	0,00255	0,92183	0,00255	0,53198	0,07448	0,66763	0,07448	**0,46420**
Sintético2_9	0,00205	1,36649	0,00205	1,36649	0,04730	0,79114	0,04730	0,79114
Sintético2_10	0,01144	0,58835	0,01144	0,58835	**0,18950**	0,52709	**0,18950**	0,52709

Tabela 6.100 D e DB do MBSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético2_1	0,12199	0,53398	0,12199	0,53398	0,16305	0,52926	0,16305	0,52926
Sintético2_2	0,04522	0,46315	0,04522	0,46315	0,18950	0,46733	0,18950	0,46733
Sintético2_3	0,02877	0,48044	0,02877	0,48044	0,18950	0,48308	0,18950	0,48308
Sintético2_4	0,08938	0,47569	0,08938	0,47569	0,04831	0,48964	0,04831	0,48964
Sintético2_5	**0,16976**	0,46254	**0,16976**	0,46254	**0,16976**	0,47013	**0,16976**	0,47013
Sintético2_6	0,12199	0,52289	0,12199	0,52289	0,16305	0,52113	0,16305	0,52113
Sintético2_7	0,06274	0,83931	0,03671	0,43603	0,13725	0,70937	0,04846	**0,45929**
Sintético2_8	0,03516	0,46972	0,03516	0,46972	0,18950	0,47494	0,18950	0,47494
Sintético2_9	0,12199	0,53398	0,12199	0,53398	0,16305	0,52926	0,16305	0,52926
Sintético2_10	0,02926	0,80523	0,02926	0,80523	0,12882	0,67326	0,12882	0,67326

Tabela 6.101 D e DB do TTSAS para cada um dos conjuntos de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético2_1	0,88043	0,54778	0,04639	0,41704	0,11943	0,59551	0,11951	0,40725
Sintético2_2	0,04482	0,48777	0,04482	0,48777	0,06587	0,50295	0,06587	0,50295
Sintético2_3	1,36899	**0,26222**	1,36899	**0,26222**	0,57836	0,46657	0,57836	0,46657
Sintético2_4	1,36899	0,38400	1,36899	0,38400	0,57836	0,47122	0,57836	0,47122
Sintético2_5	**1,38177**	0,37290	**1,38177**	0,37290	0,57836	0,47690	0,57836	0,47690
Sintético2_6	0,88043	0,54778	0,04639	0,41704	0,11943	0,59712	0,11951	0,40716
Sintético2_7	0,88043	0,54778	0,04639	0,41704	0,11943	0,59593	0,11951	0,40733
Sintético2_8	0,05847	0,80414	0,04639	0,41858	0,09873	0,54475	0,11951	0,40798
Sintético2_9	0,07627	0,53826	0,07627	0,53826	0,05951	0,49169	0,05951	0,49169
Sintético2_10	0,88043	0,54778	0,04639	0,41704	0,11943	0,59559	0,11951	0,40725

Tabela 6.102 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético2* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,01155	0,80749	0,08263	0,55869	0,78210	0,50404
M	0,01155	0,76851	0,08002	0,51836	0,44728	**0,41319**
R	0,12798	0,63377	0,15418	0,53474	0,24369	0,53382
MR	0,12996	0,61272	0,14530	0,50973	**0,24580**	0,44463

6.4.4 SINTÉTICO3

Nas tabelas 6.103 a 6.111 são apresentados os resultados da VE e dos índices D e DB para os três algoritmos, BSAS, MBSAS e TTSAS, em que os valores dos parâmetros fornecidos para os dois primeiros algoritmos foram: (1) número máximo de grupos: 5, Θ_1: 1 e *Close*: 0,5; e para o terceiro algoritmo foram: (1) Θ_1: 0,8, Θ_2: 2 e *Close*: 0,5.

Os valores da VE mostram um bom desempenho dos algoritmos na maioria dos conjuntos de dados do Sintético3, incluindo os casos em que houve melhoria com a utilização das estratégias de refinamento.

Para os conjuntos em que os métodos não apresentaram bons resultados, a característica do Sintético3, que contém dois pares de grupos muito próximos (veja Figura 6.3), não foi um fator que influenciou tanto nos resultados (como esperado); quanto aqueles já discutidos (ordem dos dados e valor de parâmetro). Nos três algoritmos, BSAS (Tabela 6.103), MBSAS (Tabela 6.104) e TTSAS (Tabela 6.105), nos dez conjuntos de dados, o que interferiu no resultado do agrupamento não foi a proximidade de dois pares de grupos, mas sim principalmente os valores de parâmetros de entrada. Considerando o conjunto Sintético3_3 (BSAS), se ao invés de usar Θ_1 = 1 usar Θ_1 = 1,5, os percentuais de erros caem de: SR=94%, M=88,8, R=60,8% e MR=60% para SR=50,4%, M=36,4, R=28% e MR=21,2%. A Tabela 6.106 mostra a média dos resultados para os quatro esquemas.

Tabela 6.103 VE do BSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético3_1	0,8	0,8	**0**	**0**
Sintético3_2	0,8	0,8	**0**	**0**
Sintético3_3	94	88,8	60,8	60
Sintético3_4	60	60	59,6	59,6
Sintético3_5	80	80	64	64
Sintético3_6	56,8	50,4	59,6	39,6
Sintético3_7	65,6	65,6	59,6	59,6
Sintético3_8	75,2	75,2	63,6	63,6
Sintético3_9	76	60	26,4	42,4
Sintético3_10	83,2	64,4	83,2	40

Tabela 6.104 VE do MBSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético3_1	91,6	20,4	90,4	38,4
Sintético3_2	90,8	90,8	89,2	89,2
Sintético3_3	69,2	69,2	70	70
Sintético3_4	93,2	93,2	91,2	91,2
Sintético3_5	90	21,6	89,6	38,8
Sintético3_6	92	21,6	90	39,2
Sintético3_7	68	68	68,8	68,8
Sintético3_8	92	**20,4**	89,6	38,4
Sintético3_9	66	66	83,6	83,6
Sintético3_10	88,4	88,4	88,8	88,8

Tabela 6.105 VE do TTSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento. M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR	M	R	MR
Sintético3_1	23,2	23,2	50	50
Sintético3_2	24	24	48,8	48,8
Sintético3_3	89,6	89,6	94,4	94,4
Sintético3_4	87,2	87,2	96	96
Sintético3_5	24,8	24,8	56	56
Sintético3_6	23,2	23,2	48	48
Sintético3_7	22,8	22,8	53,6	53,6
Sintético3_8	23,2	23,2	48	48
Sintético3_9	54	54	76,4	76,4
Sintético3_10	**19,2**	**19,2**	46,4	46,4

Tabela 6.106 Média de erro VE do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Algoritmo	SR	M	R	MR
BSAS	59,24	54,60	47,68	42,88
MBSAS	84,12	55,96	85,12	64,64
TTSAS	**39,12**	**39,12**	61,76	61,76

Tabela 6.107 VE do K-MEANS para cada um dos conjuntos de dados *Sintético3*.

Conjuntos de dados	SR
Sintético3_1	72,40
Sintético3_2	71,60
Sintético3_3	89,20
Sintético3_4	60,00
Sintético3_5	80,00
Sintético3_6	88,80
Sintético3_7	88,40
Sintético3_8	60,00
Sintético3_9	80,00
Sintético3_10	**54,80**

Os valores dos índices D e DB obtidos usando o TTSAS em todos os conjuntos de dados (Tabela 6.110) indicam uma boa estrutura nos agrupamentos obtidos, inclusive na média (ver Tabela 6.111). Isso, entretanto, não garante que houve a formação exata dos cinco grupos em todos eles.

Tabela 6.108 D e DB do BSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético3_1	1,98534	0,30051	1,98534	0,30051	**2,19972**	0,30068	**2,19972**	0,30068
Sintético3_2	1,98534	0,30051	1,98534	0,30051	**2,19972**	0,30068	**2,19972**	0,30068
Sintético3_3	0,60320	1,29421	0,60320	1,01909	0,72728	0,66740	0,72269	0,55894
Sintético3_4	0,05216	0,98784	0,05216	0,98784	0,10773	0,50625	0,10773	0,50625
Sintético3_5	0,06037	1,01885	0,06037	1,01885	0,09142	0,60484	0,09142	0,60484
Sintético3_6	0,01145	1,30123	0,00313	0,75833	0,06507	0,38469	1,04025	**0,28660**
Sintético3_7	0,00310	1,06386	0,00310	1,06386	0,09142	0,50994	0,09142	0,50994
Sintético3_8	0,06876	0,82811	0,06876	0,82811	0,09142	0,59871	0,09142	0,59871
Sintético3_9	0,00600	1,11807	0,00600	0,77168	0,01491	0,53448	0,01491	0,33602
Sintético3_10	0,00416	1,07466	0,00416	0,80683	1,04025	0,50265	1,04025	0,31808

Tabela 6.109 D e DB do MBSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético3_1	0,09142	0,68413	0,09142	0,49807	0,01247	0,77094	0,01247	0,58502
Sintético3_2	0,09142	0,64262	0,09142	0,64262	0,01247	0,70471	0,01247	0,70471
Sintético3_3	0,11978	0,54353	0,11978	0,54353	0,11815	**0,51459**	0,11815	**0,51459**
Sintético3_4	0,09142	0,58428	0,09142	0,58428	0,01247	0,66964	0,01247	0,66964
Sintético3_5	0,09142	0,67687	0,09142	0,50446	0,01247	0,72809	0,01247	0,58942
Sintético3_6	0,00754	0,68057	0,00754	0,50658	0,01955	0,74740	0,01955	0,57229
Sintético3_7	0,09142	0,62490	0,09142	0,62490	0,01247	0,69885	0,01247	0,69885
Sintético3_8	0,09142	0,65805	0,09142	0,49726	0,01247	0,72909	0,01247	0,58418
Sintético3_9	**1,16494**	0,58411	**1,16494**	0,58411	1,03392	0,66062	1,03392	0,66062
Sintético3_10	0,09142	0,60746	0,09142	0,60746	0,01247	0,67322	0,01247	0,67322

Tabela 6.110 D e DB do TTSAS para cada um dos conjuntos de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+reassignment*.

Conjuntos de dados	SR		M		R		MR	
	D	DB	D	DB	D	DB	D	DB
Sintético3_1	**3,35835**	0,53119	3,35835	0,53119	2,67027	0,58302	2,67027	0,58302
Sintético3_2	**3,35835**	0,57252	3,35835	0,57252	2,93761	0,53273	2,93761	0,53273
Sintético3_3	**3,35835**	0,55195	3,35835	0,55195	2,57430	0,56284	2,57430	0,56284
Sintético3_4	2,70489	**0,42696**	2,70489	**0,42696**	2,89752	0,50266	2,89752	0,50266
Sintético3_5	2,86129	0,49172	2,86129	0,49172	2,77152	0,52935	2,77152	0,52935
Sintético3_6	**3,35835**	0,55430	3,35835	0,55430	2,77152	0,54381	2,77152	0,54381
Sintético3_7	2,53861	0,49686	2,53861	0,49686	2,54090	0,56965	2,54090	0,56965
Sintético3_8	**3,35835**	0,55430	3,35835	0,55430	2,77152	0,54602	2,77152	0,54602
Sintético3_9	3,21419	0,50674	3,21419	0,50674	2,35705	0,61733	2,35705	0,61733
Sintético3_10	**3,35835**	0,54524	3,35835	0,54524	2,57430	0,56208	2,57430	0,56208

Tabela 6.111 Média do D e DB do BSAS, MBSAS e TTSAS para o conjunto de dados *Sintético3* considerando os quatro esquemas. SR: sem refinamento, M: *merge*, R: *reassignment* e MR: *merge+ reassignment*.

Esquema de resultado	BSAS		MBSAS		TTSAS	
	D	DB	D	DB	D	DB
SR	0,47799	0,92878	0,19322	0,62865	**3,14691**	0,52318
M	0,47716	0,78556	0,19322	0,55933	**3,14691**	0,52318
R	0,66289	0,49103	0,12589	0,68971	2,68665	0,55495
MR	0,75995	**0,43207**	0,12589	0,62525	2,68665	0,55495

6.5 Considerações Finais

Este capítulo apresentou os experimentos realizados em dez conjuntos de dados (cada um com seus dez conjuntos criados) buscando mostrar o desempenho no processo de agrupamento de pontos de dados realizado pelos três algoritmos discutidos no Capítulo 3 e, também, evidenciar a relevância, nos resultados, do uso de: (1) três estratégias de

refinamento: *merge*, *reassignment* e a combinação das duas, (2) duas técnicas de pré-processamento de dados: remoção do dado e a substituição do valor ausente do atributo e (3) Validação externa, bem como dois índices de validação interna: Dunn e Davies-Bouldin.

Os experimentos foram projetados com vistas a coletar resultados que permitissem uma avaliação empírica dos algoritmos bem como da colaboração das técnicas e estratégias implementadas.

Os resultados permitem afirmar que os três algoritmos são fortemente dependentes da ordem dos dados e, também, de uma escolha de valores de parâmetros que promova um bom agrupamento. A escolha dos parâmetros deve ser empiricamente determinada por meio de tentativas e erros e, portanto, é difícil uma escolha ótima sem qualquer informação sobre o domínio de dados.

Durante os experimentos também foi possível observar que os índices de validação interna, de certa forma e, para alguns casos, podem colaborar indicando um valor para número conveniente de grupos no agrupamento a ser gerado (ou mesmo para o valor de limiar). No entanto, esse fato depende das características de formação do agrupamento do conjunto de dados (e.g., grupos bem separados, proximidade de grupos, grupos que não estão bem definidos, etc.) e obviamente do algoritmo de agrupamento utilizado. Contrário, entretanto, a essas observações e também ao descrito em (Liu *et al*. 2010), os autores em (Halkidi *et al*. 2001) descrevem que o índice de Dunn não apresenta qualquer tendência com relação ao número de grupos. Portanto, para a especificação de tal fato, via validação interna, ainda é necessária uma investigação mais detalhada, principalmente, sobre quais formatos de grupos podem ser favoráveis (ou não) a ele por meio de validação interna.

Capítulo 7. Conclusões

Neste Capítulo são apresentadas as conclusões deste trabalho. A Seção 7.1 resume os principais pontos levantados e investigados na pesquisa realizada, bem como as principais contribuições desta dissertação. Na Seção 7.2 são discutidas as conclusões derivadas dos experimentos conduzidos, evidenciando o desempenho e as limitações dos algoritmos de agrupamento estudados neste trabalho, bem como dos esquemas utilizados e das validações de resultados. Na Seção 7.3 é apresentado um conjunto de possíveis atividades que podem ser iniciadas, em continuidade ao trabalho desenvolvido e descrito nesta dissertação.

7.1 Principais Pontos Investigados e Contribuições desta Pesquisa

Este trabalho de pesquisa investigou uma família de três algoritmos sequenciais de agrupamento em aprendizado de máquina não supervisionado: *Basic Sequential Algorithmic Scheme* (BSAS), e dois outros dele derivados chamados MBSAS (*Modified Basic Sequential Algorithmic Scheme*) e TTSAS (*Two-Threshold Sequential Algorithmic Scheme*). O trabalho contemplou também uma pesquisa subjacente da área em questão por meio da investigação de: (1) estratégias de refinamento, (2) técnicas de pré-processamento de dados e (3) validação de resultados.

A família dos três algoritmos sequenciais de agrupamento considerados neste trabalho tende a gerar agrupamentos compactos com forma esférica ou elipsoidal, dependendo da medida de distância usada. Os três algoritmos compartilham algumas características, tais como: a necessidade de um ou alguns passos e de limiares de dissimilaridade definidos pelo usuário (Θ para BSAS e MBSAS; Θ_1 e Θ_2 para TTSAS). Os valores de limiar determinam a distância máxima que um ponto de dado deve estar a partir do centróde de um grupo e ainda ser considerado como parte do grupo. O número de grupos não é inicialmente dado, mas para os dois primeiros algoritmos (BSAS e MBSAS) deve ser previsto um número máximo de grupos (q) definido pelo usuário; TTSAS para o valor de q é inicialmente definido como N (ou seja, o número de pontos de dados de entrada do conjunto de dados). No entanto, para esses algoritmos existem alguns fatores que podem influenciar diretamente nos resultados de agrupamento, tais

como: (1) a ordem em que os pontos de dados são processados e (2) os valores dos parâmetros de entrada. Tanto (1) quanto (2) podem influenciar na quantidade de grupos criados (desnecessários ou inferiores a um número apropriado) e, consequentemente na alocação correta dos dados.

Além disso, e com o objetivo de investigar a possibilidade de melhorias nos resultados de agrupamento (ou mesmo contornar os fatores que os influenciam), este trabalho também destaca que os três algoritmos podem ser melhorados em situações, em que: (1) o agrupamento resultante tem dois grupos que são suficientemente próximos para serem unidos em um único grupo e (2) existe a possibilidade de tratar a sensibilidade à ordem em que os dados são processados pelos algoritmos (embora não tão crítico para o TTSAS). Para isso, foi examinado que uma maneira de lidar com o problema (1) é através da implementação, como um processo de pós-agrupamento, de um procedimento *merge* que junta grupos considerados próximos o suficientes e, uma forma de lidar com o problema (2), é a implementação de um processo de pós-agrupamento *reassignment*, que reatribui os dados a um grupo mais próximo do qual foi alocado (se existir tal grupo mais próximo).

Este trabalho também abordou dois aspectos subjacentes à área pesquisada: (1) o pré-processamento de dados, como um processo que antecede o uso de técnicas de aprendizado, com vistas a tratar os dados disponibilizados ao aprendizado, realizado através de dois possíveis métodos para tratamento de valores ausentes: remoção do dado e substituição do valor ausente do atributo, e (2) o processo de validação de resultados obtidos, no contexto de algoritmos de agrupamento, que auxiliam tanto na busca de um agrupamento conveniente quanto na determinação de valores adequados para determinados parâmetros de algoritmos sequenciais, são eles: validação externa e validação interna de Dunn e de Davies-Bouldin.

Dessa forma este trabalho contribui com um estudo sobre as possibilidades de utilização e de desempenho dos algoritmos sequenciais de agrupamento em AM não-supervisionado, incluindo pré-processamento, estratégias de refinamento e validação de resultados. Além do embasamento teórico associado à área de pesquisa de AM não-supervisionado (discutido nos Capítulos 1 e 2) e dos métodos investigados (detalhados nos Capítulos 3 e 4) que subsidiaram esta pesquisa, no Capítulo 6 e conclusivamente na

Seção 7.2 deste Capítulo é possível evidenciar, através dos experimentos conduzidos e das análises derivadas deles, a relevância dos métodos propostos nesta pesquisa no processo de agrupamento de dados. Foi visto que, para a maioria dos conjuntos de dados utilizados, os algoritmos apresentaram bom desempenho (incluindo as estratégias e pré-processamento utilizados), bem como ajudaram a indicar os principais fatores que podem influenciar nos resultados de agrupamento. A Seção 7.2 discute as conclusões derivadas dos experimentos e destaca o desempenho e as limitações das propostas investigadas neste trabalho.

7.2 Conclusões dos Experimentos

Nos experimentos apresentados no Capítulo 6 foram utilizados seis conjuntos de dados a partir do repositório da UCI (UCI Repository 2013) e quatro sintéticos (i.e., artificialmente criados com foco em conjunto de dados cujos grupos fossem visualmente identificáveis por seres humanos). Foram utilizados os algoritmos BSAS, MBSAS e TTSAS em quatro diferentes esquemas: sem refinamento (SR), usando apenas *merge* (M), usando apenas *reassignment* (R) e utilizando ambos, *merge* e *reassignment* (MR), considerando três validações do agrupamento final: validação externa, índice de Dunn e indice de Davies-Bouldin. Para o K-Means foram apresentados apenas os resultados relativos à SR e validação externa (VE). Os conjuntos de dados utilizados têm um número variável de pontos de dados, todos descritos por atributos numéricos (incluindo a informação do ‘grupo’ a qual cada dado pertence) e, para viabilizar a metodologia seguida nos experimentos, cada um teve seus pontos de dados embaralhados de maneira a mudar sua posição no conjunto, resultando ao final dez conjuntos de dados para cada conjunto (domínio) experimentado. Nos experimentos a distância Euclidiana foi usada para medir a dissimilaridade entre pontos de dados.

Os resultados obtidos indicam que, no geral, os três algoritmos apresentaram um bom desempenho com relação aos agrupamentos obtidos (exceto para o conjunto Breast Tumor). Situações em que os resultados não foram tão satisfatórios para os conjunto de dados, foram, talvez, provocadas por alguns fatores como a ordem dos dados e valores de parâmetros.

Independente do algoritmo utilizado, o fato é que entre os dez conjuntos de cada domínio utilizado, pelo menos boa parte deles teve poucos (ou aceitáveis) erros de

alocação (alguns com até 0%) conforme os resultados analisados nas tabelas. É importante destacar que para muitos casos as estratégias de refinamento foram fundamentais para a melhoria dos resultados de agrupamentos e, neste estudo, a estratégia *reassignment* e *merge+reassignment* foram as mais eficientes quando usadas com o BSAS e com o MBSAS, e as estratégias *merge* e *merge+reassignment* quando usadas com o TTSAS.

A manutenção dos mesmos valores de parâmetros em experimentos relacionados a cada conjunto de dados (i.e. seis do UCI Repository e quatro artificiais) pode ter trazido problemas. Parâmetros fixos usados em dez conjuntos de dados idênticos, a menos da ordem de seus dados, podem ter interferido negativamente.

Dessa forma, durante os experimentos também foram verificados (por meio de tentativas) diferentes valores de parâmetros de entrada para aqueles conjuntos que não apresentaram bons resultados na validação e, com esses 'novos' valores, muitos deles obtiveram resultados de validação bem melhor, como pode ser verificado no Capítulo 6. Os experimentos permitiram também observar que a ordem em que os dados são apresentados, em geral, influencia marcadamente o desempenho dos algoritmos, independentemente, em alguns casos, do valor de parâmetro utilizado. Já o K-Means se mostrou bastante sensível ao conjunto de centróides inicialmente escolhidos, fato evidenciado em alguns casos em que, para um mesmo conjunto de dados e em pelo menos três execuções do K-Means, os resultados foram bem diversos.

Somando a essas observações, foi verificado que os fatores discutidos também podem determinar quando uma estratégia de refinamento pode colaborar ou não no resultado obtido. Dois casos podem ser observados: (1) se em um agrupamento for criada uma quantidade de grupos considerada ideal, os procedimentos *reassignment* e *merge+reassignment* podem ser os mais indicados, caso contrário, (2) ocorrerá junções de grupos e, assim, o *merge* (consequentemente também o *merge+reassignment*) é o procedimento que pode contribuir. O fato é que para *merge+reassignment*, se considerado o caso (1) acima, *reassignment* é o procedimento que determina o bom desempenho por meio das retribuições dos dados e o *merge* não faz nenhuma mudança no agrupamento dos dados, ou seja, não une grupos. O caso (2) pode acontecer de serem ambos ou qualquer um dos procedimentos (*merge* e/ou *reassignment*) capaz de gerar

mudanças no agrupamento. Neste caso se o *merge* criar uma quantidade de grupos considerada ideal (ou quase ideal), *reassignment* pode também 'ajudar' reatribuindo os dados deslocados aos possíveis grupos corretos (já que a quantidade de grupos foi modificada pelo *merge*); se o *merge* resultou em, por exemplo, apenas um único grupo, *reassignment* não fará nenhuma ação quando considerado.

Por fim, com o objetivo de validar a qualidade dos agrupamentos gerados pelos algoritmos e verificar o quanto a ordem de apresentação dos dados e os valores de parâmetros de entrada influenciaram nas estruturas de agrupamentos, este trabalho utilizou três medidas de validação, sendo uma de validação externa (VE) e outras duas de validação interna, índices de Dunn (D) e índices de Davies-Bouldin (DB).

A validação externa e a validação interna podem ser consideradas as duas principais categorias de validação de agrupamento, e a principal diferença é se a informação externa é usada ou não para a validação de agrupamento. A validação interna depende apenas das informações dos dados e avaliam a qualidade de uma estrutura de agrupamento sem considerar as informações externas. Os resultados da VE podem, de certa maneira, contribuir (em algumas situações) para indicar um valor referência para os índices de validação interna.

As medidas de validação interna (independentemente de VE) podem ser usadas para escolher, além do algoritmo de agrupamento com melhor desempenho, o número ideial de grupos (mais uma vez entra na situação por 'tentativas' e, portanto, como visto na Seção 6.5 do Capítulo 6, isto é válido para alguns casos e ainda é necessária uma investigação empírica mais detalhada sobre este fato), sem qualquer informação adicional (se esta informação adicional existir, a VE pode ter a mesma função). Na prática, a informação externa, como os grupos (classes) a que cada dado pertence, muitas vezes não está disponível. Portanto, na situação em que não há nenhuma informação externa disponível, as medidas de validação interna são as únicas opções para a validação de agrupamento. Os detalhes das análises dos resultados estão apresentados no Capítulo 6 e mostram para cada conjunto de dados como esses problemas ocorreram e as verificações realizadas.

7.3 Possíveis Atividades como Trabalho Futuro

Algumas sugestões e pretensões de possíveis atividades e de novas ideias que surgiram durante esta pesquisa são apresentadas a seguir.

A sequência deste trabalho pode ser direcionada para:

(1) a implementação de um procedimento para estimativa do número de grupos, cujas ideias estão descritas em (Theodoridis & Koutroumbas 2009). Relacionar este procedimento a um estudo (comparativo) com os índices de validação interna que, como visto, podem avaliar agrupamentos gerados com diferentes números de grupos (e assim podendo determinar o número de grupos ideal) ou mesmo avaliar qualquer outro parâmetro do algoritmo de agrupamento. Nessa proposta de estudo seriam analisados os resultados do procedimento de estimativa do número de grupos *versus* a quantidade de 'tentativas' da validação interna em busca de um número de grupos ideal (já que muitas vezes quanto melhor for o valor de índice de validação interna obtido, mais se aproxima do ideal o número de grupos estimado). Essa ideia permite experimentos em conjunto de dados sem informação adicional (i.e. número de grupos);

(2) uma investigação mais detalhada sobre quais formatos de grupos não são bons para validação interna, considerando os algoritmos investigados;

(3) articulada ao descrito em (1), outra proposta é a criação de um algoritmo (envolvendo ou não validação), para a estimativa dos valores dos limiares dos algoritmos;

(4) investigar as vantagens do BSAS implementado como uma rede neural, como proposto em (Theodoridis & Koutroumbas 2009);

(5) implementar o índice Estatística Γ Modificada por Hubert.

Referências

Abdella M. and Marwala T. 2005. The use of genetic algorithms and neural networks to approximate missing data in database. In: Computational Cybernetics. IEEE 3rd International Conference, pp. 207-2012.

Acuña E. and Rodrigues C. 2004. The Treatment of Missing Values and Its Effect on Classifier Accuracy. Proceedings of the Meeting of the International Federation of Classification Societies (IFCS), Illinois Institute of Technology, Chicago, USA, pp. 639-647.

Allison, P. D. 2001. Missing Data. Thousand Oaks, CA: Sage.

Anderberg, M. R. 1973. *Cluster Analysis for Applications*. Academic Press, New York, NY.

Ahmadi, N. and Berangi, R. 2008. A basic sequential algorithmic scheme approach for classification of modulation based on neural network, In: *Proceedings of the IEEE International Conference on Computer and Communication Engineering Kuala Lumpur*, Malaysia, pp. 565-569.

Arlot, S. and Celisse, A. 2010. A survey of cross-validation procedures for model selection. *Statistics Surveys online journal*, (4), pp. 40-79.

Bertini Jr., J. R. and Nicoletti, M.C. 2008. A constructive neural network algorithm based on the geometric concept of barycenter of convex hull. In: Rutkowski, L., Tadeusiewiza, R., Zadeh, L.A., Zurada, J. (eds) *Computacional Inteligence: Methods and Applications*, (1), Academic Publishing House EXIT, Warsaw, pp. 1-12.

Bertini Jr., J. R. and Nicoletti, M.C. 2008a. MBabCoNN - a multiclass version of a constructive neural network algorithm based on linear separability and convex hull, Proceedings of *ICANN* (2), pp. 723-733.

Bishop, C. M. 1999 Neural networks for pattern recognition, *Great Britain*: Oxford University Press.

Blend D. and Marwala T. 2008. Comparison of Data Imputation Techniques and their Impact. 2008, arXiv:0812.1539. Disponível em: < http://arxiv.org/ftp/arxiv/papers/0812/0812.1539.pdf>.

Blum, A. and Mitchell, T. M. 1998. Combining labeled and unlabeled Data with Co-Training, Proceedings of *COLT*, *Madison* – Wisconsin, pp. 92-100.

Bottou, L. and Bengio, Y. (1995). Convergence Properties of the KMeans Algorithm, *Advances in Neural Information Processing Systems*, 7, MIT Press, Denver, pp. 585-592.

Bouckaert, R. R., Frank, E., Hall, M., Kirkby, R., Reutemann, P., Seewald, A. and Scuse, D. 2012. WEKA Manual for Version 3-6-8. University of Waikato, Hamilton, New Zealand. Disponível em: <http://ufpr.dl.sourceforge.net/project/weka>.

Carpenter, G. A. and Grossberg, S. 1987. ART2: Self-organization of stable category recognition codes for analog input patterns, *Applied Optics*, (26), pp. 4919-4930.

Clark P.; Niblet T. 1988. The CN2 induction algorithm, *Machine Learning Journal*, 3 (4), pp. 261-283.

Dai, D. B., Tang, S.L. and Xiong, Y. (2010). Clustering Algorithms Based on Global and Local Similarity. *Journal of Software Sequence*, 21(4), pp. 702-717.

Das, N. 2003. Hedge fund classification using K-means clustering method. In: *9th International Conference in Computing in Economics and Finance*, University of Washington, Seattle. Disponível em: < http://depts.washington.edu/sce2003/Papers/284.pdf >.

Davies, D. L., Boudin, D. W. 1979. A cluster separation measure. *IEEE Trans. Pattern Anal. Machine Intelligence*, 1(4), pp. 224-227.

Diday, E. and Simon, J. J. 1976. Clustering Analysis. *In: Digital pattern Recognition* (K. S. Fu, ed.), pp. 47-94.

Dubes, R. C. 1993. Cluster analysis and related issues. *In: Handbook of Pattern Recognition & Computer Vision*, C. H. Chen, L. F. Pau and P. S. P. Wang, Eds. World Scientific Publishing Co., Inc., River Edge, NJ, pp. 3–32.

Dunn, J. 1974. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, (4), pp. 95-104.

Fahlman, S. and Lebiere, C. 1991. The Cascade-correlation learning architecture, *CMU-CS-90-100, CMU-CS-90-100 School of Computer Science Carnegie Mellon University*, Pittsburgh – USA, pp. 524-532.

Fisher, D. H. 1987. Knowledge acquisition via incremental conceptual clustering, *Machine Learning*, (2), pp. 139-172.

Frank, A. and Asuncion, A. 2010. UCI Machine Learning Repository, Irvine, CA: University of California, *School of Information and Computer Science*, http://archive.ics.uci.edu/ml.

Fu, L., Yang, M., Braylan, R. and Benson, N. 1993. Real-time adaptive clustering of flow cytometric data, *Pattern Recognition*, (26), n. 2, pp. 365-373.

Gallant S. I. 1990. Perceptron Based Learning Algorithms. *IEEE Transaction on Neural Networks*, (1), pp. 179-191.

Gallant S. I. 1993. Neural Network Learning and Expert Systems. *Cambridge MA. MIT Press*.

Giraud-Carrier, C., Martinez, T. 1995. ILA: Combining inductive learning with prior knowledge and reasoning, *University of Bristol, Department of Computer Science*. *Also issued as* ACRC-95: CS-003, pp. 1-17.

Guan, B. X., Bhanu, B., Thakoor, N. S., Talbot, P. and Lin, S. 2013. Automatic Cell Region Detection By K-Means With Weighted Entropy. In: *IEEE 10th International Symposium on Biomedical Imaging*: From Nano to Macro San Francisco, CA, USA, pp. 418-421.

Haldiki, M., Batistakis, Y. and Vazirgiannis, M. 2001. On Clustering Validation Techniques. *Journal of Intelligent Information Systems*, pp. 107-145.

Halkidi, M. , Batistakis, Y. and Vazirgiannis, M. 2002(a). Cluster Validity Methods: Part I. *SIGMOD* Record 31(2), pp. 40-45.

Halkidi, M. , Batistakis, Y. and Vazirgiannis, M. 2002(b). Clustering Validity Checking Methods: Part II. *SIGMOD* Record 31(3), pp. 19-27.

Halkidi, M. and Vazirgiannis, M. 2001. Clustering Algorithms and Validity Measures. *SSDBM*, pp. 3-22.

Hall, A. V. 1967. Methods for demonstrating resemblance in taxonomy and ecology, Nature, (214), pp. 830-831.

Han, J; Kamber, M. 2006. Data Mining: Concepts and Techniques. Elsevier.

Hubert L. and Arabie P. 1985. Comparing partitions. In. Journal of Classification, v. 2, pp. 193-218.

Ichino, M. and Yaguchi, H. 1994. Generalized Minkowski metrics for mixed feature-type data analysis. *IEEE Trans. Syst. Man Cybern.* 24, pp. 698–708.

Jain, A.K. and Dubes, R.C. 1988. Algorithms for Clustering Data, *Prentice Hall*.

Jain, A. K., Murty, M. N. and Flynn, P. J. 1999. Data clustering: a review. *ACM Computing Surveys*, (31), n.3, pp. 264-323.

Keller, F. 2012. Evaluation: connectionist and statistical language processing. Universitat des Saarlandes, Computerlinguistik. Disponível em: < http://www.coli.uni-saarland.de/~crocker/Teaching/Connectionist/lecture11_4up.pdf >. [Acessado 02 outubro 2012].

Kohavi, R.; Provost, F. 1998. Glossary of terms: special issue on applications of machine learning and the knowledge discovery Process, *Kluwer Academic Publishers*, Boston, pp. 271-274.

Kovács, F., Legány, C. and Babos, A. 2001. Cluster Validity Measurement Techniques. Technical report, Department of Automation and Applied Informatics, Budapest University of Technology and Economics, Budapest, Hungary, pp-107-145.

Kovács, F., Legány, C. and Babos, A. 2005. Cluster Validity Measurement Techniques. *Proceeding: AIKED'06 Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases*, pp. 388-393.

Liu, Y., Gao, B. and Zhang X. 2011. An Improved Sequential Clustering Algorithm. In: *Third International Conference*, AICI 2011, Taiyuan, China, Proceedings, Part I, pp. 444-449.

Mao, J. and Jain, A. K. 1996. A self-organizing network for hyperellipsoidal clustering (HEC). *IEEE Trans. Neural Netw. 7*, pp. 16–29.

Maurya R., Singh S., Gupta P.R. and Sharma M. K. (2011). Road Extraction Using K-Means Clustering and Morphological Operations. In: *International Journal of Advanced Engineering Sciences and Technologies*, v. 5, n. 2, pp. 290-295.

Mei, X. and Lei, Z. 2008. Using modified basic sequential clustering for background reconstruction, *Information Technology Journal*, (7), n. 7, pp. 1037-1042.

Mitchell, T. M. 1997. Machine Learning, USA: *McGraw-Hill*.

Mitchell, T. M. 1982. Generalization as Search. *Elsevier: Artificial Intelligence*, (18), n. 2, pp. 203-226.

Muggleton, S. 1987. Duce, an Oracle Based Approach to Constructive Induction. *In: IJCAI-87* - Kaufmann, pp. 287-292.

Muggleton, S. and Buntine, W. 1988. Machine Invention of First-order predicates by inverting resolution. *In: Proccedings of the Fifth International Conference on Machine Learning* – Kaufmann, pp. 359-352.

Muggleton, S. and Feng, C. 1993. Efficient induction of logic programs, *Proc. of the First Conference on Algorithmic Learning Theory*, Tokyo, Japan.

Michalski, R. S. 1969. On the Quasi-Minimal Solution of the General Covering Problem. *Fifth International Symposium on Information Processing*, A3: 125-128.

Nicoletti, M. C. 1994. Ampliando os limites do aprendizado indutivo de máquina através das abordagens construtiva e relacional, *Ph. D.* , IFSC-USP.

Nicoletti, M.A., Magalhães, J.F. and Nicoletti, M.C. 1998. O uso do sistema CN2 na indução de conhecimento em domínio farmacotécnico, *RT-DC 005/98, UFSCar/DC*, São Carlos, 45 pgs.

Nicoletti, M.C., Bertini Jr., Elizondo, D., Franco, L. and Jerez, J. M. 2009 Constructive neural network algorithms for feed forward architectures suitable for classification tasks, *in: Constructive Neural Networks, Studies in Computational Intelligence*, Chapter 1, Springer-Verlag, (258), pp. 1-23.

Osuna, R. G. 2012. Inteligente Sensor System, *Wright State University*. Disponível em: < http://research.cs.tamu.edu/prism/lectures/iss/iss_l13.pdf >. [Acessado 05 outubro 2012].

Pal N.R. and Biswas, N.R. (1997). Cluster validation using graph theoretic concepts. In. *Pattern Recognition*, n. 30, pp. 847–857.

Quinlan, J. R. 1990. Learning from relational data, *Proc. of The 4th Australian Joint Conference on Artificial Intelligence*, World Scientific, pp. 38-47.

Quinlan, J. R. 1990. Learning logical definitions from relations, *Machine Learning*, (5), pp. 239-266.

Quinlan, J. R. 1986. Induction of decision trees, *Machine Learning*, (1), pp. 81-106.

Quinlan, J. R. 1993. Programs for Machine Learning. *Morgan Kaufmann Publishers*, Inc. USA: Editorial Office - 2929 Campus Drive, Suite 260, San Mateo, CA 94403.

Raedt, L. 1992. Interactive Theory Revision: an Inductive Logic Programming Approach. *Academic Press*.

Rosenberg, C., Hbert, M. and Shneiderman, H. 2005. Semi-supervised self-training of object detection models, Robotics Institute, paper 374, pp. 29-36.

Rouveirol, C. 1992. Extensions of inversion of resolution applied to theory completion, In: Muggleton S. (ed.), *Inductive Logic Programming*. London: Academic Press.

Sammut, C. and Banerji, R. B. 1986. Learning Concepts by Asking Questions. *In: Michalski, R.; Carbonnel, J.; and Mitchell, T. editors, Machine Learning: An Artificial Intelligence Approach*, (2), Kaufmann, pp. 167-192.

Schlimmer, J. C. and Fisher, D. 1986. A case study of incremental concept induction, *Proceedings of the Fifth National Conference on Artificial Intelligence. Philadelpha*, PA: Morgan Kaufmann, pp. 496-501.

Steinbach, M., Karypis, G., and Kumar, V. (2000). A comparison of document clustering techniques. *KDD workshop on text mining*, pp. 1-20.

Tan, P-N., Steinbach, M. and kumar, V. (2005). Introduction to Data Mining. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA.

Tatiraju S. and Mehta A. (2008). Image Segmentation using k-means clustering, EM and Normalized Cuts. In: Machine Learning Winter. Disponível em: <http://www.ics.uci.edu/~dramanan/teaching/ics273a_winter08/projects/avim_report.pdf>

Theodoridis, S.; Koutroumbas, K. 2009. Pattern Recognition, 4th ed., USA: *Elsevier*.

Trahanias, P.; Scordalakis, E. 1989. An efficient sequential clustering method, *Pattern Recognition*, (22), n. 4, pp. 449-453.

UCI Repository. 2013. Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science. Disponível em: < http://archive.ics.uci.edu/ml>

Utgoff, P. E. 1988. ID5: an incremental ID3, *Proceedings of the Fifth International Conference on Machine Learning*, Ann Arbor, MI: Morgan Kaufman, pp. 107-120.

Utgoff, P. E. 1989. Improved training via incremental learning, Proceedings of the Sixth International Workshop on Machine Learning. Ithaca, NY: Morgan Kaufmann, 362-365.

Xiao, M and Han, C. Z. 2007. Background subtraction algorithm based on online clustering. *Moshi Shibie yu Rengong Zhineng*, 20, pp. 35-41.

Zhang W., Yang Y. and Wang Q. 2012. A Comparative Study of Absent Features and Unobserved Values in Software Effort Data. *In: International Journal of Software Engineering and Knowledge Engineering*, Beijing, China, pp. 185-202.

Anexo

Trabalho aceito para apresentação no *13th International Conference on Intelligent Systems Design and Applications* (ISDA 2013) e publicado nos anais da conferência pelo *IEEE*.

Eduardo M. Real <eduardomreal@gmail.

ISDA13 notification for paper 12

10 mensagens

ISDA13 <isda13@easychair.org> 20 de outubro de 2013
Para: "E. M. Real" <eduardomreal@gmail.com>

Dear Author (s),

12:
The Impact of Refinement Strategies on Sequential Clustering Algorithms:

Congratulations! On behalf of the ISDA 2013 Technical Program Committee, we are pleased to inform that your paper has been accepted for presentation at the 13th International Conference on Intelligent Systems Design and Applications (ISDA 2013) to be held in Malaysia and for publication in the conference proceedings published by IEEE. Each paper was send to at least five independent reviewers and based on their recommendations your paper was accepted. We expect to have about 50 technical presentations during the conference. If you have any questions related to the conference, please dont directly reply to this email but send your email to the appropriate contacts (include your paper ID in all correspondences).

This email provides you with all the information you require to complete your paper and submit it for inclusion in the proceedings. Please read carefully and here are the steps you must follow:

13th International Conference
Intelligent Systems Design and Applications
Universiti Putra Malaysia, Malaysia 08-10, December 2013

The Impact of Refinement Strategies on Sequential Clustering Algorithms

Maria do Carmo Nicoletti, Eduardo Machado Real and Osvaldo Luiz de Oliveira

FACCAMP & UFSCar-DC S. Carlos, SP, Brazil UEMS & FACCAMP Nova Andradina, MS, Brazil FACCAMP C. L. Paulista, SP, Brazil

carmo@cc.faccamp.br eduardomreal@gmail.com osvaldo@faccamp.br

ABSTRACT

Sequential clustering algorithms have been characterized as fast and straightforward methods which produce, as result, a single clustering. They have the drawback of being dependent on the order in which data patterns are input to the algorithm and, generally, produce compact and spherical clusters. The focus of the work is a group of sequential algorithms which includes the Basic Sequential Algorithmic Scheme (BSAS) and two of its variations, the MBSAS and the TTSAS. The paper investigates refinement strategies which aim to improve the performance of the three sequential algorithms based on two processes: merge and reassignment. Results from experiments conducted in various data domains (from UCI and synthetic) are presented and a comparative analysis is given as evidence of the benefits of sequential clustering algorithm coupled with a refinement

The Impact of Refinement Strategies on Sequential Clustering Algorithms

Maria do Carmo Nicoletti
FACCAMP & UFSCar-DC
S. Carlos, SP, Brazil
carmo@cc.faccamp.br

Eduardo Machado Real*
UEMS & FACCAMP
Nova Andradina, MS, Brazil
eduardomreal@gmail.com
*corresponding author

Osvaldo Luiz de Oliveira
FACCAMP
C. L. Paulista, SP, Brazil
osvaldo@faccamp.br

Abstract – Sequential clustering algorithms have been characterized as fast and straightforward methods which produce, as result, a single clustering. They have the drawback of being dependent on the order in which data patterns are input to the algorithm and, generally, produce compact and spherical clusters. The focus of the work is a group of sequential algorithms which includes the *Basic Sequential Algorithmic Scheme* (BSAS) and two of its variations, the MBSAS and the TTSAS. The paper investigates refinement strategies which aim to improve the performance of the three sequential algorithms based on two processes: merge and reassignment. Results from experiments conducted in various data domains (from UCI and synthetic) are presented and a comparative analysis is given as evidence of the benefits of sequential clustering algorithm coupled with a refinement procedure.

Keywords – clustering, sequential clustering algorithms, sequential clustering with merge and reassignement procedures.

I. INTRODUCTION

Clustering algorithms can be considered one of the most successful approaches in data mining and are the most popular among the unsupervised learning algorithms available. The main goal of clustering methods is to organize a given set of data patterns into groups (clusters) and the main strategy employed is to group them based on their similarity. The idea underneath a clustering procedure can be a convenient way to organize the available data and help solving problems in many different areas of knowledge such as medicine, engineering, biology and so on.

Differently to the way supervised algorithms work, the data patterns input to clustering algorithms have no pre-assigned classes. Therefore, the basic learning strategy employed by clustering algorithms is to discover similarities and differences in the input patterns so to gather them into groups aiming, at the end of the clustering process, to disclose some sort of organization of the given set of input patterns. The groups of patterns produced by a clustering algorithm, however, can be approached as 'categories (or classes)' and can be further used to categorize new patterns. A clustering learning strategy infers a set of groups the input patterns may be organized into, which would reveal similarities and differences between them as well as help to derive conclusions about them.

In the literature several clustering algorithms, supported by various mathematical and statistical formalisms, can be found such as Expectation Maximization (EM) [1], K-means [2], the COBWEB, an incremental system for hierarchical conceptual clustering [3], DBScan [4], Dynamic clustering [5] and Chameleon [6], to name just a few. Due to the many available algorithms and, also, to the different strategies adopted by them, various taxonomies based on several different criteria can be found in the literature [7] [8]. They all attempt to organize and gather in groups those algorithms that share some relevant characteristics and adopt similar strategies. In [8], for instance, clustering algorithms are approached divided into the following major categories: (1) sequential (2) hierarchical (3) based on cost function optimization and (5) others. Halkidi and co-workers in [9] suggest that clustering algorithms can be classified according to: (1) the type of data input to the algorithm, (2) the clustering criterion defining the similarity between data patterns and (3) the theoretical framework they employ. Adopting as criteria the way clusters are defined, Jain and co-workers [8] group algorithms into four main categories: (1) partitional, (2) hierarchical, (3) density-based and (4) grid-based. No matter the taxonomy, their main categories invariably have subcategories, in an attempt to accommodate other aspects of the algorithms, such as type of data (crisp or fuzzy or categorical).

Algorithms characterized as sequential, according to [8] are considered simple, fast and produce a single clustering as result. The input data to be clustered can be presented to a sequential algorithm once or a few times and, as a rule, the final result depends on the order in which the data are presented. The work described in this paper approaches sequential based clustering as a family of three algorithms namely: the *Basic Sequential Algorithmic Scheme* (BSAS) and two of its variation named MBSAS (*Modified Basic Sequential Algorithmic Scheme*) and TTSAS (*Two-Threshold Sequential Scheme*). The BSAS is presented in [8] as a generalization of the proposal described in [10] and the two others (MBSAS and TTSAS) can be considered variations of the BSAS.

As discussed in details in Section II, there are a few drawbacks in the three algorithms which can, somehow, be minimized by two simple strategies referred to as *merge* and *reassignment* which can be used as two post-clustering procedures. They have been proposed for fixing certain situations not addressed by the algorithms. There is, for instance, the possibility that during the clustering process, two groups are formed relatively close to each other and, depending on the degree of their proximity, it could be advantageous to join them into a single group, a task that can accomplished by a procedure that merges them together, as described in [11]. Another situation that commonly can

happen during the clustering process is the strong dependence of results upon the order in which the data patterns are processed by the algorithm. Once a pattern is assigned to a group, it remains in that group for the rest of the processing. This sometimes can be quite inconvenient, since other groups may be created which would be more suitable for a pattern to belong to instead of the one initially considered. So, in an attempt improve this aspect of the algorithms, a procedure referred to as reassignment is considered, as suggested [8].

The work described in this paper enlarges the refinement options by using only one of the two procedures or then a combination of them. The remainder of the paper is organized as follows. Section 2 discusses the shared characteristics of the three algorithms as well as those that are particular to each of them. Section 3 describes the refinement process and details the two procedures involved namely, the merge and the reassign procedures, presenting their pseudocodes. Section 4 presents the experiments and the analysis of the inferred clustering by learning schemes where one or both refinement strategies have been used. It also presents the results of the obtained clustering, using an external evaluation process Section 5 summarizes the work done and presents some conclusions.

II. The Sequential Based Clustering Family

The three-member family of sequential clustering algorithms considered in this work tend to generate compact clusters which are spherical or ellipsoidal shaped, depending on the distance measure used. The three algorithms share some characteristics, such as: the need for one or a few steps and user-defined thresholds of dissimilarity (Θ for BSAS and MBSAS; Θ_1 and Θ_2 for TTSAS). The threshold values determine the maximum distance a data pattern should be from the center of a group and still be considered as part of the group. The number of groups is not initially given but for the first two algorithms (BSAS and MBSAS) a user-defined maximum allowable number of groups should be provided (q); for TTSAS the value of q is initially set to N (i.e., the number of data patterns given as input).

In what follows the notation used in this work is introduced. The input data for a clustering algorithm, i.e., the set of data patterns to be clustered will be represented by $CP = \{E_1, E_2, \ldots, E_N\}$ ($|CP| = N$) where each E_i ($1 \leq i \leq N$) is described as a vector of M attributes ($AT=\{A_1, A_2, \ldots, A_M\}$). Pattern E_i is given by $[E_{i1}, E_{i2}, \ldots, E_{iM}]$ where E_{i1} is a possible value of attribute A_1, E_{i2} is a possible value of attribute A_2, and so on.

The three algorithms return a single clustering C, which can be seen as a set of sets. Each set belonging to C is a group of clustering G. So, the output of each clustering algorithm is noted by $G = \{G_1, G_2, \ldots, G_Z\}$ - depending on the algorithm there will be a threshold for the maximum number of groups allowed in the final clustering i.e., a limit on the value of Z). The three algorithms have been implemented considering each group in a given clustering $G = \{G_1, G_2, \ldots, G_Z\}$ being represented by a pattern referred to as *group representative*, implemented as the mean vector of the data patterns that belong to the group.

A. BSAS

Algorithm 1 shows a high level pseudocode of BSAS. Since the main focus of this work is on the refinement process and, also due to the limit upon the number of pages, the MBSAS and TTSAS high level pseudocodes are not presented; their high level description can be found in [8]. At each iteration the BSAS considers a data pattern $E_i \in CP$ and, depending on the distance between E_i and the representative of each of the created group so far, two possible courses of action are possible: (a) to include E_i in one of the existing groups or then (b) to create a new group having E_i as its representative. If situation (a) holds, the representative of the group E_i has been included to, needs to be updated. The order in which patterns are processed by BSAS has a direct influence on the final outcome of the algorithm in relation to both, the number of groups created as well as the patterns that belong to each group.

The values of q and Θ also have a role in the final clustering obtained. If the value assigned to Θ is too small, unnecessary groups can be created and if it is too large, a small number (below the appropriate number) will be created. As stressed in [8], improper choice of Θ may lead to meaningless clustering results. If the maximum number of groups allowed for grouping (q) is not previously established, BSAS creates as many groups as appropriate.

B. MBSAS

As shows Algorithm 1, patterns from CP are considered one at a time and, for each one considered, the BSAS either (a) includes it into an existing group or then (b) creates a new group having the pattern as a member. The decision to do (a) or (b) is made based on the existing groups created so far and it is final i.e., not reviewed by the algorithm. It may happen though, that a pattern can be assigned to group which is not as suitable as another group that might be created later, for instance.

The MBSAS tries to overcame this problem by considering the set of patterns twice. The first time it creates a certain number of groups, each having a single pattern; the second time it assigns loose patterns to the most suitable group among those created before. So, the MBSAS can be viewed as a version of BSAS in which the group formation process is refined. Both algorithms share the same two parameters i.e., the limit on the value of dissimilarity (Θ) and the a limit on the maximum number of possible groups to be created (q).

The two main steps of MBSAS can be summarized as follows: (1) groups containing a single pattern are created (to a maximum of q) and (2) loose patterns are then assigned to their closest group. Similarly to the BSAS, the MBSAS is also sensitive to the order in which data patterns are processed and, also, instead of using a similarity measure, can be adapted for using a dissimilarity measure.

```
procedure BSAS
Input: CP: {E1, …, EN} {input set }(1 ≤ i≤ N)
M: number of attributes describing each Ei (1 ≤ i≤ N)
Θ: threshold of dissimilarity
q : maximum allowed number of groups
Output: G = {G1, G2, ..., GZ} (1 ≤ Z ≤ q), resulting clustering
1. begin
2. G ← ∅
3. group_counter ← 1
4. G_group_counter ← {E1}    {first group in G}
5. n_group_counter ← 1
6. G ← G ∪ G_group_counter
7. for i ← 2 to N do
8.   begin
9.    smallest_dist ← distance(E1,G1)
10.   group_smallest_dist ← 1
11.   for j ← 2 to group_counter do
12.     if d(Ei,Gj) < smallest_dist then
13.       begin
14.         smallest_dist ← distance(Ei,Gj)
15.         group_smallest_dist ← j
16.       end
17.   if (distance(Ei,G_group_smallest_dist) > Θ)
18.     then
19.       if (group_counter < q)      {creating a new group}
20.         then begin
22.                 group_counter ← group_counter + 1
23.                 G_group_counter ← {Ei}
24.                 n_group_counter ← 1
25.                 G ← G ∪ G_group_counter
26.               end
27.         else send_message(' q too small')
28.     else begin
30.           G_group_smallest_dist ← G_group_smallest_dist ∪ {Ei}
31.           n_group_smallest_dist ← n_group_smallest_dist + 1
32.         end
33.end.
34. return(G)
```

Algorithm 1. Pseudocode of BSAS – a clustering of the input data having q (maximum) groups is created, using a user-defined dissimilarity value Θ.

C. *TTSAS*

As mentioned before both, BSAS and MBSAS, are strongly influenced by the order in which data patterns are processed as well as by the supplied user-defined value for Θ. The third algorithm of the sequential family, named TTSAS (*Two-Threshold Sequential Scheme*), was proposed with the intent to minimize these influences. It does that by using two threshold parameters, Θ_1 and Θ_2 (with $\Theta_2 > \Theta_1$) which define a 'gray' region in the input space [12] which allows the establishment of following three rules: (1) if the dissimilarity value of a data pattern E_x to its nearest group G_k is less than Θ_1 then E_x is assigned to G_k; (2) if the dissimilarity value of a data pattern E_x to its nearest group G_k is great than Θ_2 then a new group is created and E_x is assigned to it; otherwise, (3) a dissimilarity value of a data pattern E_x to its nearest group G_k is greater than Θ_1 and small than Θ_2 is interpreted as a signal of uncertainty and the assignment of pattern E_x is postponed. TTSAS is always, at least, almost as computationally expensive as BSAS and MBSAS, since it generally requires at least two passes on the input data. Furthermore, since the assignment of a pattern to a group is deferred until sufficient information is gathered, the algorithm turns out to be less sensitive to the order the patterns are processed. As happens with the two previous algorithms, different choices of dissimilarity measure between a pattern and a group lead to different results. The TTSAS also favors compact clusters, when groups are represented by representatives.

III. THE REFINEMENT PROCESS

Although MBSAS and TTSAS are considered improvements of BSAS, the three algorithms could still be further improved in situations such as: (1) when the resulting clustering has two groups that are sufficiently close to be merged into one and (2) the recurrent sensitivity to the order in which the data pattern are processed by the algorithms (although not that critical for the TTSAS). One way to deal with problem (1) is by implementing, as a post-clustering process, a procedure that merges groups considered close enough, as proposed in [13] and described in Algorithm 2. As suggested in [8], a way to deal with problem (2) is by implementing a post-clustering process which reassigns patterns which would be considered misplaced; the pseudocode of such procedure is presented in Algorithm 3.

Figure 1 and the corresponding Table I shows an example where two groups, G_1 and G_4, are merged since they are 'close enough' i.e., the distance between their representatives is smaller than the value of parameter Close (=2.5).

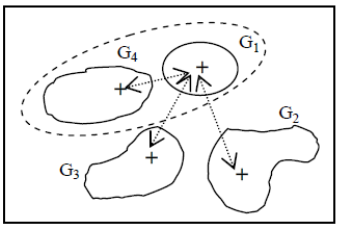


Figure 1. Clustering G={G_1, G_2, G_3, G_4} and the merging of the two closest groups - see Table 1.

TABLE I. CLUSTERING G={G_1, G_2, G_3, G_4} AND PAIRWISE DISTANCES BETWEEN GROUPS' REPRESENTATIVES.

G_i	G_j	d_{ij}	value(d_{ij})
G_1	G_2	d_{12}	6
G_1	G_3	d_{13}	3
G_1	G_4	d_{14}	2
G_2	G_3	d_{23}	5
G_2	G_4	d_{24}	7
G_3	G_4	d_{34}	4

Procedure *merge* in Algorithm 2 requires as input: (1) a clustering obtained using any of the algorithms discussed

before, noted as G: $\{G_1, \ldots, G_Z\}$ and (2) a user-defined value for parameter Close, which allows to identify, in a clustering, groups that are close enough to be merged. Procedure *indices_2groups_smaller_distance*() identifies in the clustering the two groups which are the closest to each other among all the groups. If they are close enough (according to Close), they are merged into a single group and then they are renamed (as well as the others); the process is repeated until no such two groups are detected.

```
procedure merge
Input:
G: {G1, …, GZ} {output of BSAS, MBSAS or TTSAS}
Close: maximum distance allowed between two groups in a
clustering, that would, still, qualify them for merging.
Output: G = {G1, G2, ..., GV} {resulting clustering from the
merge process applied to the original clustering G = {G1, G2,
..., GZ} (1 ≤ V ≤ Z)}

1. begin
2.  continue ← true
3.  while continue do
4.   begin
5.    indices_2groups_smaller_distance(G,i, j)
6.    if distance(Gi,Gj) < Close
7.     then begin
8.          Gi ← merge(Gi,Gj)
9.          G ← remove(G, Gj)
10.         RGi ← update(RGi) {update representative of Gi}
11.         for k ← j+1 to Z do
12.           begin
13.            rename(Gk,Gk-1)
14.            Z ← Z – 1
15.           end
16.        end
17.   else continue ← false
18. end
19. return(G)
```

Algorithm 2. Pseudocode of procedure *merge*, which expects as input a clustering given by G = $\{G_1, \ldots, G_Z\}$ and a user-defined parameter Close, representing how close two groups should be, to be merged.

Procedure *reassignment* (Algorithm 3) requires as input: (1) a clustering obtained using any of the algorithms discussed before, noted as G: $\{G_1, \ldots, G_Z\}$ and (2) the initial set of data patterns.

For each initial pattern E_i ($1 \leq i \leq N$) procedure *closest* identifies its closest group representative and assigns the pattern to it (for the vast majority of patterns the assignment will not produce any change). In the next step the group representatives are updated.

As a side effect of the reassignment procedure, there is a chance of a group ending up with no pattern. The last for command in Algorithm 3 removes it from the clustering. As expected, the resulting clustering from the procedure reassignment can have a smaller number of groups than the input clustering.

```
procedure reassignment
Input:
CP: {E1, E2, ..., EN} {initial set of N patterns}
G: {G1, …, GZ} {output of BSAS, MBSAS or TTSAS}
Output: G = {G1, G2, ..., GV} {result from the reassignment
process - V ≤ Z}

1. begin
2.  for i ← 1 to N do begin
3.                closest(Ei,G,j)
4.                group(Ei) ← j
5.               end
6.  for j ← 1 to Z do  begin
7.                Gj ← {Ei ∈ CP | group(Ei) = j}
8.                update_representative(Gj)
9.               end
10. for j ← 1 to Z do if is_empty(Gj) then
11.                begin
12.                  G ← G – {Gj}
13.                  Z ← Z – 1
14.                end
15. end
16. return(G)
```

Algorithm 3. Pseudocode of procedure *reassignment*, which transfers misplaced patterns to groups closer to them. It expects as input a clustering given by G = $\{G_1, \ldots, G_Z\}$ and the initial set of patterns.

IV. Experiments and Analysis of Results

The clustering methods (Section II) and the proposed refinements (Section III) used in the experiments described in this section have been implemented in Delphi (version 7–build 4453). They are part of a software system called SEQ_CLUSTER which has been developed aiming at experimenting with sequential clustering techniques. The system runs under a Microsoft Windows platform.

The SEQ_CLUSTER functional architecture has been organized into three main modules: (1) the pre-processing module, in charge of translating the input data into a unique standard format (expected by the clustering module) as well as removing data instances with problems, such as missing attribute values, (2) the data set generator module, responsible for automatically creating synthetic data sets based on user´s specifications and (3) the clustering module, containing the implementations of the clustering methods and the refinement procedures i.e., merge and reassignment.

For the experiments a total of 5 data sets have been chosen; three have been downloaded from the UCI repository [14] and two (synthetic1 and synthetic2) have been artificially created having in mind perceptually identifiable groups (see Figure 2 and Figure 3, respectively). Table I presents a summary of their main characteristics. The data sets have a variable number of data patterns and they all have their data patterns described by numerical attributes (for the clustering experiments the Euclidean distance has been used to measure dissimilarity). The UCI chosen data sets are typically data sets for supervised learning tasks – the reason for that was also to be able to evaluate the obtained results using an external validation index.

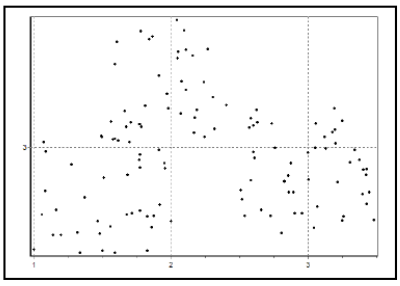


Figure 2. Data set synthetic1.

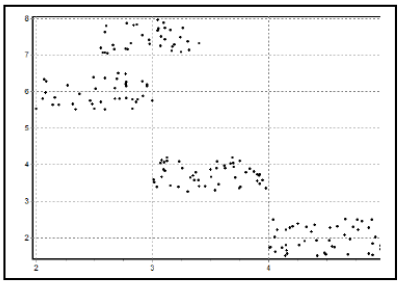


Figure 3. Data set synthetic2.

TABLE II. DATA SETS SUMMARY. #NP: NO. OF PATTERNS; #NA: NO. OF ATTRIBUTES, #NC: NO. OF CLASSES AND #NP/CLASS: NO. OF PATTERNS PER CLASS. OBS: THE ORIGINAL HEART DATA SET HAS 13 ATTRIBUTES.

Data sets	#NP	#NA	#NC	#NP/CLASS
ecoli	336	7	8	143/cp
				77/im
				52/pp
				20/om
				5/omL
				2/imL
				2/imS
				35/imU
heart	270	6	2	120/0
				150/1
iris	150	4	3	50/setosa
				50/versicolor
				50/virginica
synthetic1	125	2	3	50/a
				35/b
				35/c
synthetic2	170	2	4	40/a
				45/b
				50/c
				35/d

Each data set from Table II was input to each of the three clustering algorithms namely BSAS, MBSAS and TTSAS. For each data set and for each clustering algorithm, four results were obtained, taking into account the four possible schemes: (1) no refinement, (2) only using merge, (3) only using reassignment and (4) using both, merge and reassignment.

Table III presents the parameters values used and they have been kept unchanged throughout the experiments. It is important to mention that the process of finding a suitable set of parameter values is, *per se*, not a trivial task. Also, during the experiments it could be observed the sensitiveness of the three algorithms to parameter values – very small changes in parameter values would produce significant changes in the obtained clustering. The commonly practiced approach [16], also employed in this work, was to try to find good values for the parameters before running the experiments; once these values were found, they remained fixed during the runs.

TABLE III. PARAMETER VALUES. C: NO. CLUSTERS, Θ, Θ_1 AND Θ_2: THRESHOLDS, CLOSE: MAXIMUM CLOSENESS ALLOWED.

Data sets	BSAS and MBSAS			TTSAS		
	C	Θ	Close	Θ_1	Θ_2	Close
ecoli	8	0.5	0.3	0.3	0.8	0.5
heart	2	100	50	100	200	150
iris	3	2.5	1	2.5	5	2
synthetic1	3	1	0.5	1	2	0.5
synthetic2	4	1	0.5	1	2	1.5

The obtained results were evaluated using an external validation (EV) process that quantifies the number of data patterns correctly assigned (using the original class which is part of the description of each pattern, for the UCI files and the perceptually identified groups, in both synthetic data sets).

Tables IV, V and VI show the results of EV for the three algorithms, BSAS, MBSAS and TTSAS, respectively, taking into account the four schemes, WR: without refinement, M: merge, R: reassignment and MR: merge + reassignment. The tables show the final results for each data set, obtained by averaging the incorrect results in a repeated (10 times) approach (and representing by the corresponding %); patterns in each data set were shuffled giving rise to 9 other data sets. For each data set, the 9 associated data sets share the same patterns but in different order.

The numbers reported in the three tables are evidence that the best results were obtained when a sequential algorithm (BSAS, MBSAS or TTSAS) is coupled with a refinement procedure (i.e., M, R or MR). When BSAS was used (Table IV) that was the case for all the five data sets; the use of merge alone, however, has not improved results in any data set.

TABLE IV. EV OF BSAS. WR: WITHOUT REFINEMENT; M: MERGE; R: REASSIGNMENT AND MR: MERGE+REASSIGNMENT. BOLD FACED NUMBERS REPRESENT LOWEST % OF INCORRECTLY GROUPED PATTERNS.

Datasets	WR	M	R	MR
ecoli	51.60	54.50	**45.68**	63.65
heart	60.00	55.81	55.74	**53.70**
iris	23.40	34.13	22.13	**21.47**
synthetic1	24.48	24.96	**11.20**	14.80
synthetic2	55.76	42.11	**24.35**	28.35

When using MBSAS (Table V) and TTSAS (Table VI) the same tendency as before can be observed; the best results were shared among the schemes coupled with refinements. Considering the five data sets and the three clustering

algorithms without refinement (WR) it is interesting to note that (1) BSAS had the best performance in ecoli (although not good) and iris; (2) MBSAS in heart (although not good) and synthetic1 (3) TTSAS in synthetic2 (a good one). Considering that the two variants (MBSAS and TTSAS) have been proposed with the intention to refine the original BSAS their performances in the five data sets do not reflect their purposes. Contrary to what happened when using BSAS, the merge refinement coupled with TTSAS had best performances (although three of them not good) in four out of five data sets. The experiments are not conclusive in relation to the best refinement strategy but their results support their use.

TABLE V. EV OF MBSAS. SAME CONVENTION AS TABLE IV.

Datasets	WR	M	R	MR
ecoli	79.46	**57.89**	72.92	73,85
heart	44.33	44.33	44.15	**43.59**
iris	23.73	23.73	**22.20**	**22.20**
synthetic1	16.56	18.56	**12.96**	14.80
synthetic2	56.94	64.94	54.11	**50.47**

TABLE VI. EV OF TTSAS. SAME CONVENTION AS TABLE IV.

Datasets	WR	M	R	MR
ecoli	72.62	**56.21**	87.61	66.52
heart	67.07	**45.11**	88.89	46.41
iris	61.00	48.13	48.67	**40.67**
synthetic1	36.80	**33.28**	60.72	38.40
synthetic2	13.29	**10.29**	27.17	12.05

V. CONCLUSIONS AND FUTURE WORK

This paper discusses unsupervised learning experiments where two refinement strategies, suitable to be coupled to sequential clustering algorithms, aiming at improving results, are discussed and empirically evaluated. The sequential algorithms used were the BSAS and two of its variants, MBSAS and TTSAS. The three algorithms and the two refinement strategies were implemented in Delphi as the software system SEQ_CLUSTER.

Experiments were conducted using 5 data sets, three of them downloaded from the UCI repository and two artificially created. The whole experiment involved the stand-alone use of the three sequential algorithms (BSAS, MBSAS and TTSAS) as well as the use of each of them coupled with one possible scheme of refinement (i.e., merge, reassignment and both). Although the experiments were conducted in a limited number of data, their results seems to support coupling a refinement strategy as a post-processing phase to clustering, in an attempt to refine results.

The experiments also confirmed a common problem that happens when using parameterized algorithms: the three algorithms are very sensitive to parameter values – very small changes in parameter values provoke significant changes in their results. As the algorithms are also sensitive to the order of the data patterns, the experiments tried to promote a fair environment: for each data set each scheme was run 10 times each time considering a particular order among the patterns. Although the performance rates obtained, measured by the external index, are not as good as expected, they can contribute for a better understanding of the three algorithms as well as their proposed refinements and, maybe be the post-processing refinement strategies could, somehow, be embedded in the algorithm and taken into account during the clustering process, a possibility to be further investigated. Also the experiments will be extended to consider a larger number of data domains.

ACKNOWLEDGMENTS

Authors would like to express their special thanks of gratitude to Faculdade Campo Limpo Paulista (FACCAMP) for supporting this research work. The first author would like also to thank to CNPq for the research scholarship received.

REFERENCES

[1] A. P. Dempster and D. B. Rubin, "Maximum likelihood form incomplete data via the EM algorithm," J. of the Royal Statistical Soc., 1977, pp. 1-38.

[2] J. B. MacQueen, Some methods for classification and analysis of multivariate observations," Proc. of the Sysmposium on Mathematical Statistics and Probability, 1967, pp. 281-297.

[3] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," Machine Learning, v. 2, no. 2, 1987, pp. 139-172.

[4] M. Ester, H. P. Kriegel, J. Sander and X. Xu, "A density-based algorithm for discovering clusters in large databases with noise," Proc. of the Second Internationl Conference on Knowledge Discovey and Data Mining, 1996, pp. 226-231.

[5] E. Diday, "The dynamic cluster method in non-hierarquical clustering," Journal Comput. Inf. Sci, 1973, pp. 61-88.

[6] G. Karypis, E.-H. Han and V. Kumar, "Chameleon: a hierarchical clustering algorithm using dynamic modeling," IEEE Computer, v. 32, no. 8, 1999, pp. 68-75.

[7] A. K. Jain, M. N. Murty and P. J. Flynn, "Data clustering: a review," ACM Computing Surveys, 1999, pp. 264-323.

[8] S. Theodoridis and K. Koutrombas, Pattern Recognition, USA: Elsevier, 2009.

[9] M. Halkidi, Y. Batistakis and M. Vazirgiannis, "On clustering validation techniques," Journal of Intelligent Information Systems, v. 17, no. 2/3, 2001, pp. 107-145.

[10] A. V. Hall, "Methods for demonstrating resemblance in taxonomy and ecology," Nature, 1967, pp. 830-831.

[11] L. Fu, M. Yang, R. Braylan and N. Benson, "Real-time adaptative clustering of flow cytometric data," Pattern Recognition, 1993, pp. 365-373.

[12] P. Trahanias and E. Scordalakis, "An efficient sequential clustering method," Pattern Recognition, 1989, pp. 449-453.

[13] L. Fu, M. Yang, R. Braylan and N. Benson, "Real-time adaptative clustering of flow cytometric data," Pattern Recognition, 1993, pp. 365-373.

[14] A. Frank and A. Assuncion, "UCI Machine Learning Repository," Irvane, CA: University of California, School of Information and Computer Science, 2010, Available: http://archive.ics.uci.edu/ml.

[15] J. Dunn, "Well separated clusters and optimal fuzzy partitions," Journal of Cybernetics, 2008, v. 4, pp. 95-104.

[16] J. Brest, B. Boskovic, M. Mernik, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," IEEE Transactions on Evolutionary Computation, 2006, v. 10, no. 6, pp. 646-657.