Incorporação de Múltiplos Representantes Auxiliares em Processos de Detecção de Agrupamentos Semi-supervisionados

Walter José da Silva

Universidade Federal de Uberlândia
Faculdade de Computação
Programa de Pós-Graduação em Ciência da Computação

Uberlândia - MG
2015

Walter José da Silva

Incorporação de Múltiplos Representantes Auxiliares em Processos de Detecção de Agrupamentos Semi-supervisionados

Dissertação de mestrado apresentada ao Programa de Pós-graduação da Faculdade de Computação da Universidade Federal de Uberlândia como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Área de concentração: Ciência da Computação

Orientador: Profª Drª Maria Camila Nardini Barioni
Coorientador: Profª Drª Sandra Aparecida de Amo

Uberlândia - MG
2015

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da UFU, MG, Brasil.

S586i
2015

Silva, Walter José da, 1989-
Incorporação de múltiplos representantes auxiliares em processos de detecção de agrupamentos semi-supervisionados / Walter José da Silva. - 2015.
128 f. : il.

Orientadora: Maria Camila Nardini Barioni.
Coorientadora: Sandra Aparecida de Amo.
Dissertação (mestrado) - Universidade Federal de Uberlândia, Programa de Pós-Graduação em Ciência da Computação.
Inclui bibliografia.

1. Computação – Teses. I. Barioni, Maria Camila Nardini. II. Amo, Sandra Aparecida de. III. Universidade Federal de Uberlândia, Programa de Pós-Graduação em Ciência da Computação. IV. Título.

CDU: 681.3

Dedico aos meus pais, Valdomiro e Marina, e aos meus irmãos.
À minha amiga, companheira e namorada, Raniele.
Dedico, principalmente, a pessoa que mais incentivou e apoiou para que eu sempre seguisse em frente, possibilitado mais essa realização em minha vida. Pai, obrigado.

Agradecimentos

À Profª. Maria Camila, por todos os ensinamentos e paciência ao longo desses últimos anos. Posso dizer que aprendi muito e que sou muito grato por tudo.

À Profª. Sandra, que com sua experiência trouxe uma visão diferenciada, o que contribuiu para enriquecer ainda mais o trabalho.

Ao Prof. Humberto, pela enorme ajuda durante esse trabalho. Nossas reuniões, meio improvisadas, fizeram uma grande diferença no desenvolvimento de todo o trabalho.

À minha namorada Raniele, pelo amor, apoio e compreensão durante todo esse tempo.

Aos meus pais, Valdomiro e Marina, por todo o suporte e incentivo para que eu pudesse conquistar mais essa etapa na minha vida.

Aos meus amigos, Bruno, Emília, Franciny, Myllene e todos os outros colegas que contribuíram de forma direta ou indireta para a realização desse mestrado.

À CAPES, Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, e a FAPEMIG pelo apoio financeiro.

"Há verdadeiramente duas coisas diferentes: saber e crer que se sabe. A ciência consiste em saber; em crer que se sabe reside a ignorância."
(Hipócrates)

Resumo

A incorporação de semi-supervisão no processo de detecção de agrupamento de dados tem sido especialmente útil quando se deseja obter uma alta consistência entre o particionamento dos dados e o conhecimento do usuário sobre a verdadeira estrutura dos dados. Nos últimos tempos, várias estratégias para detecção de agrupamentos semi-supervisionado de dados têm sido propostas. As abordagens adotadas por essas estratégias têm como objetivo guiar o processo de detecção de agrupamentos por meio do uso de restrições com os seguintes propósitos: interferindo na atribuição das instâncias ao grupo mais apropriado a cada iteração do algoritmo; ou modificando a função objetivo utilizada. Esta dissertação apresenta uma nova abordagem para incorporar semi-supervisão ao amplamente conhecido algoritmo *k-means*. Esse método de agrupamento semi-supervisionado emprega as informações de restrições na definição de múltiplos representantes auxiliares para os centróides utilizados a cada iteração do *k-means* e na geração de novos tipos de restrições que agem em *nível de protótipo*. Um processo de refinamento é desenvolvido para reduzir o número de representantes auxiliares considerados a cada centróide, sem perder a qualidade do agrupamento. Os resultados experimentais mostram o potencial da abordagem proposta para lidar com agrupamentos de diferentes formas, tamanhos e densidades.

Palavras-chave: agrupamento de dados, agrupamento semi-supervisionado, agrupamento por particionamento, restrição em nível de protótipo.

Abstract

The incorporation of semi-supervision in the cluster detection process has proved especially useful when one wants to get a high consistency between the data partitioning and the knowledge the user has about the data domain. In recent years, several strategies for semi-supervised clustering have been proposed. The approaches adopted by these strategies aim at guiding the process of cluster detection by using constraints with the following purposes: to interfere with the allocation of instances to the most appropriate cluster at each iteration of the algorithm; or to modify the objective function employed. This dissertation presents a novel approach for incorporating semi-supervision in the well-known k-means algorithm. This semi-supervised clustering method employs constraint information in the definition of multiple assistant representatives for the centroids used at each iteration of k-means and generating new types of constraints acting on prototype level. A refinement process is designed to reduce the number of assistant representatives considered for each centroid without losing the clustering quality. The experimental results show the potential of the proposed approach for dealing with clustering composed by clusters of different shapes, sizes and densities.

Keywords: data clustering, semi-supervised clustering, partitional clustering, prototype-level constraints.

Lista de ilustrações

Lista de tabelas

Lista de símbolos

n	Número de instâncias no conjunto de dados
d	Número de dimensões do conjunto de dados
k	Número de grupos
$X = \{x_1, ..., x_n\}$	Conjunto de dados
$\Pi = \{C_1, ..., C_k\}$	Conjunto de partições
x_i	Instância do conjunto de dados X
C_j	Conjunto de instâncias de um grupo
χ_l	Subconjunto de instâncias em uma componente conexa
μ_j	Centróide principal do grupo C_j
a_i	Centróide auxiliar
M_μ	Conjunto de centróides principais
M_a	Conjunto de centróides auxiliares candidatos
M'_a	Conjunto de centróides auxiliares
I_m	Conjunto de instâncias que participam de alguma restrição *must-link*
M_χ	Conjunto de componentes conexas χ
S_j	Subconjunto de centróides auxiliares pertencentes a um grupo C_j
Q_k	Conjunto de representantes do grupo C_k (μ_k e diversos $a_i \in M'_a$)
$\delta()$	Função de similaridade
$\delta_g()$	Função de similaridade agregada
R_{ml}	Conjunto de restrições *must-link* em nível de instância
R_{cl}	Conjunto de restrições *cannot-link* em nível de instância
$r_{ml}(x_i, x_j)$	Restrição *must-link* entre x_i e x_j
$r_{cl}(x_i, x_j)$	Restrição *cannot-link* entre x_i e x_j
R'_{ml}	Conjunto de restrições *must-link* em nível de protótipo
R'_{cl}	Conjunto de restrições *cannot-link* em nível de protótipo
$r'_{ml}(a_i, a_j)$	Restrição *must-link* entre a_i e a_j
$r'_{cl}(a_i, a_j)$	Restrição *cannot-link* entre a_i e a_j

ρ_i^j	*Rank* do j-ésimo algoritmo no i-ésimo conjunto de dados
$\bar{P}_j$	*Rank* médio do j-ésimo algoritmo
P_j	Soma dos *ranks* do j-ésimo algoritmo
χ_F^2	Estatística de *Friedman*
F_F	Derivação da estatística de *Friedman*
CD	Diferença crítica
z	Diferença de desempenho entre dois algoritmos

Sumário

Capítulo 1

Introdução

O objetivo da análise de agrupamentos é fornecer meios para identificar estruturas de grupos em um conjunto de dados baseando-se apenas em métricas, características intrínsecas percebidas ou similaridade entre suas instâncias (JAIN, 2010). Esta é uma das principais tarefas descritivas empregadas em conjunto com processos de mineração de dados. Quando busca-se por palavras como "*clustering*", "*summarization*" e "*association*" na biblioteca digital DBLP, encontra-se respectivamente 2236, 225 e 565 artigos somente em 2014 (DBLP, 2015). O grande número de respostas obtidas para a palavra-chave "*clustering*" mostra como essa tarefa de análise de dados é importante.

Apesar do fato de sua extensiva e rica história, a análise de agrupamentos pode ainda ser considerada uma tarefa difícil e desafiadora. Dentre as razões para isso pode-se destacar a possibilidade de se ter diferentes estruturas que descrevem um mesmo conjunto de dados e a falta de conhecimento sobre a verdadeira estrutura existente nos dados para guiar o processo de agrupamento. Essas questões são ilustradas na Figura 1.1. A partir de um conjunto de dados (Figura 1.1(a)), vários particionamentos e diferentes quantidades de partições são possíveis para esses dados, como mostram as Figuras 1.1(b), 1.1(c) e 1.1(d).

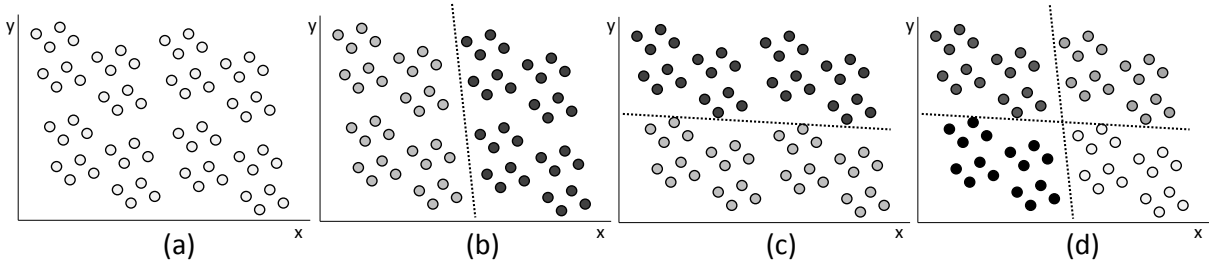

Figura 1.1 – Exemplo dos vários particionamentos possíveis para um mesmo conjunto de dados.

A fim de lidar com essas questões, diferentes campos de pesquisa têm considerado o uso de estratégias que permitem interferir, de certa forma, no processo de agrupamento,

guiando-o a um desejável ou mais adaptável particionamento dos dados. Dentre elas está a análise visual de agrupamentos, abordagens interativas com o usuário e agrupamento semi-supervisionado (BARIONI et al., 2014). Para o agrupamento semi-supervisionado, as abordagens desenvolvidas neste emergente campo de pesquisa incorporam informações adicionais sobre o conjunto de dados ao processo de agrupamento com o objetivo de impor propriedades de agrupamentos desejáveis. Essa informação adicional pode ser expressa, por exemplo, na forma de rótulos a uma pequena porção do conjunto de dados, ou restrições, as quais afirmam que duas instâncias devem ou não devem pertencer a um mesmo grupo. Exemplos de restrições em *nível de instância* que impõem que duas instâncias devem pertencer a um mesmo grupo (*must-link*), ou que duas instâncias não devem pertencer a um mesmo grupo (*cannot-link*) podem ser vistas na Figura 1.2(a). Essas restrições influenciam o particionamento dos dados para um resultado mais próximo do agrupamento esperado, veja a Figura 1.2(b).

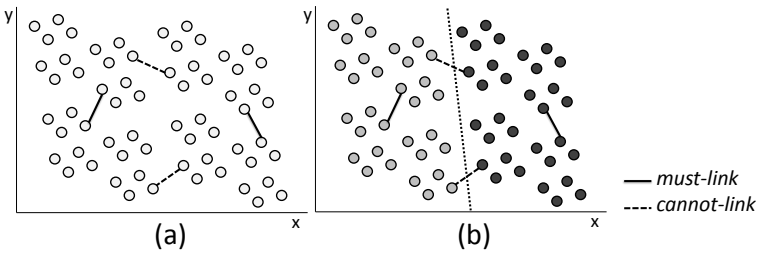

Figura 1.2 – Influência das restrições entre instâncias no particionamento resultante.

A maioria dos algoritmos de agrupamento semi-supervisionados descritos na literatura são baseados em clássicos algoritmos de particionamento de dados (WAGSTAFF et al., 2001) ou hierárquicos (ZHENG; LI, 2011). Contudo, pode-se dizer que as características de seus algoritmos são herdadas. Em geral, algoritmos de agrupamentos particionais têm melhor desempenho quando lidam com grupos de formato esférico. Por outro lado, algoritmos de agrupamentos hierárquicos são mais adaptáveis a descoberta de grupos de formas arbitrárias, embora apresentem um maior custo computacional. Com o intuito de permitir a detecção de estruturas de agrupamentos mais complexas, alguns trabalhos adotam uma estratégia que emprega múltiplos protótipos representantes por grupo (GUHA; RASTOGI; SHIM, 1998; LIU; JIANG; KOT, 2009).

O trabalho descrito nesta dissertação tem o objetivo de incorporar esta habilidade de manipular estruturas de grupos complexas em processos de agrupamento semi-supervisionados por particionamento de dados. Assim, é apresentada nesta dissertação uma nova abordagem de semi-supervisão que emprega restrições em nível de instância para derivar múltiplos representantes auxiliares para cada particionamento gerado pelo algoritmo. Além disso, as restrições entre pares de instâncias são utilizadas para gerar novos tipos de restrições, as quais atuam em *nível de protótipo* e que contribui para uma melhor uti-

lização dos múltiplos representantes. Diferentemente das restrições em nível de instância, que impõem que um par de instâncias deve ou não deve pertencer a um mesmo grupo, as restrições em nível de protótipo impõem que um par de protótipos, ou representantes, deve ou não representar um mesmo grupo.

A nova abordagem de semi-supervisão proposta foi incorporada ao amplamente conhecido algoritmo *k-means*, criando assim, o método chamado MRS-*kmeans* (*Multi-Representative Semi-supervised k-means*). Em resumo, a ideia principal do MRS-*kmeans* é fornecer meios para lidar com duas questões, como ilustra a Figura 1.3: a descoberta de uma desejável estrutura de agrupamento no conjunto de dados dentre as várias existentes, como mostra a Figura 1.3(a) com uma quantidade de grupos diferente da Figura 1.3(b); e a detecção de estruturas de agrupamento contendo grupos de diferentes formas, tamanhos e densidades.

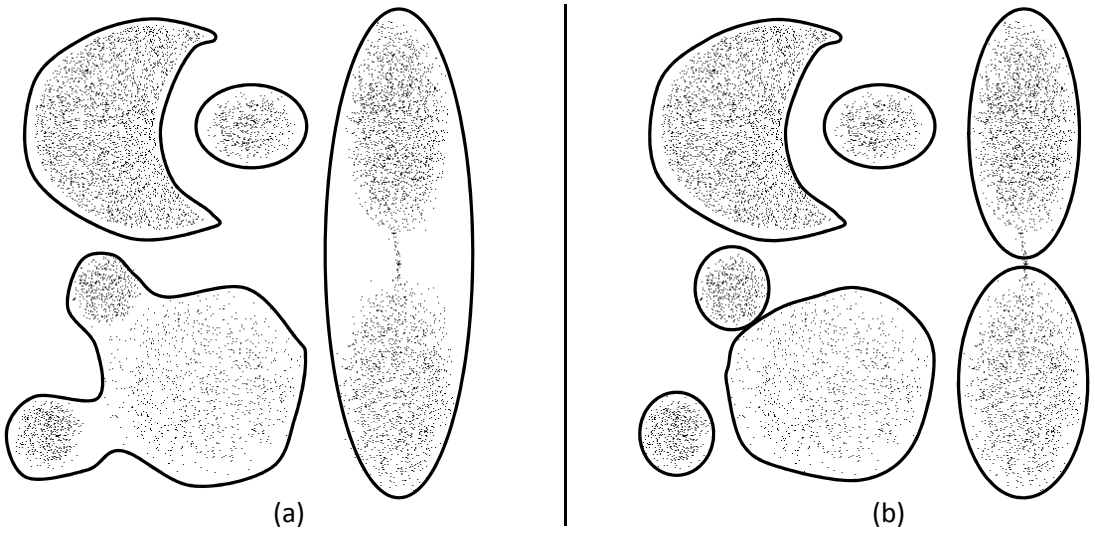

Figura 1.3 – Diferentes particionamentos para uma mesma estrutura dos dados. (a) 4 grupos. (b) 7 grupos.

1.1 Objetivos

O objetivo direto do trabalho realizado descrito nesta dissertação consistiu em abordar o estudo e o desenvolvimento de um ferramental teórico e prático que contribuísse para aprimorar a qualidade semântica de agrupamentos obtidos por métodos de detecção de agrupamentos, considerando conjuntos de dados com estruturas complexas. Para tanto, foram exploradas estratégias para:

- Transformar informações de restrições entre pares de instâncias em múltiplos representantes para cada grupo;

- ❏ Derivar as informações de restrições em nível de instância para restrições em *nível de protótipo*, as quais auxiliam a correta atribuição dos múltiplos representantes para os grupos mais adequados;
- ❏ Desenvolver meios de selecionar os representantes auxiliares mais representativos para uma detecção de agrupamentos de qualidade e de menor custo;
- ❏ Utilizar técnicas que consideram múltiplos representantes no cálculo de distância realizado na fase de atribuição das instâncias aos grupos;
- ❏ Utilizar uma metodologia de avaliação própria para algoritmos de detecção semi-supervisionada de agrupamentos.

O foco do trabalho realizado, descrito nesta dissertação, é apresentar o método desenvolvido e analisar sua eficiência com ênfase na qualidade do particionamento de dados gerado. A partir disso, tem-se meios para gerar as seguintes contribuições.

1.2 Principais Contribuições

As principais contribuições do trabalho realizado são:

- ❏ Criação de um novo método de detecção de agrupamentos semi-supervisionado por particionamento capaz de aproveitar as informações contidas em restrições entre instâncias do tipo *must-link* e *cannot-link*, gerando novas formas de conhecimento, e as utilizando para melhorar o desempenho em algoritmos com abordagens de particionamento de dados em conjuntos de dados de estruturas complexas.
- ❏ Criação de um novo tipo de informação de semi-supervisão, em forma de restrições entre os novos protótipos, que auxiliam a correta atribuição desses protótipos aos grupos mais adequados.
- ❏ Detecção aprimorada de agrupamentos em conjuntos de dados de estruturas complexas, possibilitando realizar processos de detecção de agrupamentos mais acurados em conjuntos de dados, os quais possuem estruturas de agrupamentos contendo grupos de formas, tamanhos e densidades arbitrárias. Também, realizando a detecção de agrupamentos com menor número de iterações e consequentemente retornando o particionamento dos dados com um menor custo computacional.

1.3 Organização da Dissertação

Esta dissertação está organizada em seis capítulos como mostrado a seguir:

- **Capítulo 1:** Apresenta a introdução e a motivação para realização do trabalho descrito nesta dissertação.
- **Capítulo 2:** Apresenta os conceitos fundamentais em detecção de agrupamentos, múltiplos protótipos e medida de similaridade agregada, a fim de prover uma melhor compreensão do trabalho realizado.
- **Capítulo 3:** Traz conceitos específicos ao contexto de semi-supervisão, apresentando também alguns trabalhos relacionados.
- **Capítulo 4:** Apresenta todos os detalhes da nova abordagem de detecção de agrupamentos por meio de semi-supervisão criada durante a realização do trabalho descrito aqui.
- **Capítulo 5:** Mostra os resultados dos experimentos realizados para validar a nova abordagem apresentada nesta dissertação, mostrando também o desempenho superior dessa abordagem em relação a trabalhos da literatura que utilizam de abordagens de semi-supervisão.
- **Capítulo 6:** Conclui esta dissertação com algumas considerações e propostas para possíveis trabalhos futuros. Apresenta também as publicações geradas com a pesquisa desenvolvida.
- **Apêndice A:** Traz algumas tabelas de valores críticos para avaliação estatística realizada na fase de experimentos.

Capítulo 2

Detecção de Agrupamentos

A tarefa de análise de agrupamentos em conjuntos de dados é um ramo da mineração de dados em constante evolução nos últimos tempos, e tem sido bastante utilizada em diversas aplicações, sendo algumas delas: reconhecimento de padrões, análise de dados, e processamento de imagens. Em aplicações que envolvam processamento de imagens, por exemplo, tem sido usada para melhorar a indexação de imagens para sistemas de Recuperação de Imagens Baseada em Conteúdo (CBIR, do inglês *Content-Based Image Retrieval*). O processo de agrupamento pode ser amplamente definido como a divisão de um conjunto de instâncias em determinados grupos de acordo com uma medida de similaridade, de forma que cada um desses grupos reúna instâncias que apresentem características em comum que sejam diferentes das características das instâncias representadas por grupos distintos (HAN; KAMBER, 2006; FACELI et al., 2011).

Contudo, existem várias maneiras de agrupar instâncias baseadas nas suas características. Ou seja, podem haver formas distintas de grupos para um mesmo conjunto de dados, como exemplificado pela Figura 2.4. Considerando o conjunto de dados ilustrado na Figura 2.4(a) constituído por diversas formas geométricas (instâncias), o primeiro agrupamento (Figura 2.4(b)) divide as instâncias em dois grupos pela sua forma. No segundo (Figura 2.4(c)), o agrupamento divide as instâncias pelo seu preenchimento, novamente em dois grupos. Por fim, o último agrupamento (Figura 2.4(d)) separa as instâncias em quatro grupos considerando ambas características, forma e preenchimento (FACELI et al., 2011).

Considerando as instâncias do conjunto de dados em um espaço d-dimensional, sendo d a quantidade de dimensões que representam as características desses dados, os grupos podem ser vistos como um subconjunto de instâncias que possuem uma relação de proximidade espacial, o que se refere basicamente à similaridade existente entre elas. Entretanto, o termo **grupo** não tem uma definição formal bem estabelecida, e isso se deve ao fato dos diferentes tipos de pesquisas realizadas sobre análise de agrupamentos em diversas áreas. Porém, existem algumas definições comumente usadas para o termo, que são (BARBARA, 2000):

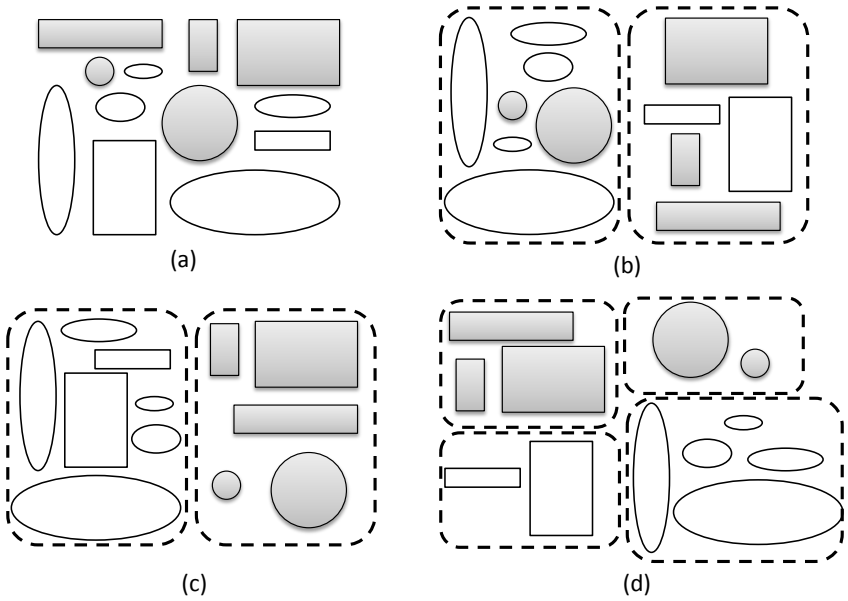

Figura 2.4 – Formas agrupadas em diferentes maneiras (FACELI et al., 2011).

- **Grupo bem definido:** É um conjunto de instâncias tal que, qualquer instância em um grupo está mais próxima a todas as outras do mesmo grupo do que qualquer outra instância que não esteja nele. Algumas vezes, um *threshold* é usado para especificar que todas instâncias em um grupo devem estar suficientemente próximas umas das outras;

- **Grupo baseado em representante:** É um conjunto de instâncias tal que, uma instância está mais próxima do "representante" de um grupo do que o representante de outro grupo. Esse representante geralmente é um **centróide**, que é a média calculada de acordo com todas as instâncias em um grupo, ou um **medóide**, a instância mais representativa do grupo;

- **Grupo contínuo:** É um conjunto de instâncias tal que, uma instância em um grupo está mais próxima a uma ou mais instâncias deste grupo do que para qualquer instância que não está nele;

- **Grupo baseado em densidade:** É uma região de instâncias de alta densidade, sendo que essas regiões são separadas por regiões de baixa densidade. Esta definição é frequentemente usada quando os grupos são irregulares ou interlaçados, e quando ruídos e *outliers* estão presentes;

- **Grupo baseado em similaridade:** É um conjunto de instâncias que são **similares** (por alguma medida de similaridade) em relação a outras instâncias que, por sua vez, não são similares ou são dissimilares. Esse tipo de grupo define um conjunto de instâncias que cria uma região com propriedade local uniforme, por exemplo, densidade ou forma.

Os detalhes sobre o processo geral de detecção de agrupamentos são apresentados no decorrer deste capítulo, o qual está organizado da seguinte forma. A Seção 2.1 descreve os detalhes inerentes ao processo de detecção de agrupamentos em um contexto geral. Na Seção 2.2 é apresentada a abordagem de detecção de agrupamentos por particionamento, tal abordagem é utilizada no trabalho descrito nesta dissertação. A Seção 2.3 apresenta alguns algoritmos que utilizam a estratégia de múltiplos protótipos para gerar agrupamentos de dados e uma estratégia de utilização dos múltiplos protótipos ao cálculo da similaridade. Por fim, a Seção 2.4 encerra com algumas considerações sobre o capítulo.

2.1 Etapas do Processo de Agrupamento

O processo de detecção de agrupamentos pode ser dividido em várias etapas, como mostrado na Figura 2.5. Por meio dessa figura, também pode-se observar o fluxo percorrido pelo conjunto de dados inicial até a identificação e validação dos grupos. As subseções seguintes descrevem em detalhes cada uma das etapas intermediárias desse processo.

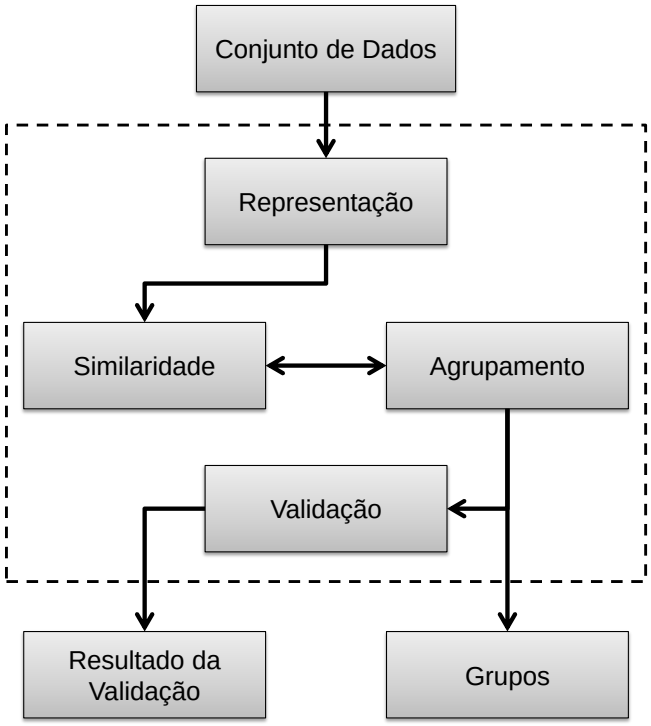

Figura 2.5 – Etapas do processo de detecção de agrupamento, adaptado de (FACELI et al., 2011).

2.1.1 Representação do Conjunto de Dados

A representação de um conjunto de dados é, de maneira geral, expressa ou abstraída como uma matriz de dados $n \times d$, com n linhas e d colunas. Nessa matriz de dados,

as linhas correspondem as instâncias do conjunto de dados e as colunas correspondem às dimensões ou atributos desse conjunto. Cada linha da matriz define um conjunto de características de uma dada instância (WITTEN; FRANK, 2005; ZAKI; MEIRA, 2014). A matriz de dados $n \times d$ pode ser ilustrada como segue:

$$X = \left(\begin{array}{c|cccc} & x^1 & x^2 & \cdots & x^d \\ \hline x_1 & x_1^1 & x_1^2 & \cdots & x_1^d \\ x_2 & x_2^1 & x_2^2 & \cdots & x_2^d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_n & x_n^1 & x_n^2 & \cdots & x_n^d \end{array}\right)$$

sendo que x_i representa a i-ésima linha, a qual é uma d-tupla dada por $x_i = \left(x_i^1, x_i^2, ..., x_i^d\right)$; e x^j representa a j-ésima coluna, a qual é uma n-tupla dada por $x^j = \left(x_1^j, x_2^j, ..., x_n^j\right)$.

Contudo, nem todos os conjuntos de dados podem estar representados na forma de matriz de dados. Conjuntos de dados mais complexos que, frequentemente, estão na forma de sequências (por exemplo, DNA e sequência de proteínas), texto, séries-temporais, imagens, áudio, vídeo e etc., podem precisar de técnicas especiais para realizar a análise. Em alguns casos, quando esses dados brutos não estiverem em um formato de matriz, cada instância desse conjunto pode ser transformada em um vetor de características, extraído por meio de algum descritor. Por exemplo, dado um conjunto de dados de imagens, pode-se criar uma matriz de dados na qual as linhas representam cada imagem e as colunas correspondem a algum conjunto de características extraídas, tal como: cor, textura ou forma. Outros tipos de atributos podem ter uma semântica específica e necessitam de algum tratamento diferenciado, como é o caso de atributos temporais e espaciais (ZAKI; MEIRA, 2014).

Em um conjunto de dados, dependendo de seu domínio, seus atributos podem ser classificados em dois tipos principais: numéricos e categóricos (WITTEN; FRANK, 2005; ZAKI; MEIRA, 2014). Isto é, sua classificação depende dos tipos de valores que esses atributos assumem, como mostrado a seguir.

- **Atributo Numérico:** os atributos numéricos podem assumir valores reais ou inteiros. Os atributos que admitem um conjunto finito ou contavelmente infinito de valores são chamados **discretos**, ao passo que, aqueles atributos que admitem qualquer valor real são chamados de **contínuos**.

- **Atributo Categórico:** os atributos categóricos assumem valores em um conjunto finito e pré-especificado de possibilidades. Geralmente, esse tipo de atributo tem o objetivo de rotular ou nomear uma determinada característica.

O conjunto de dados também pode ser classificado em escalas, indicando o significado relativo aos valores de seus atributos. As escalas dos dados podem ser divididas em:

qualitativas e quantitativas. A escala qualitativa inclui os tipos de dados nominal e ordinal, enquanto que a escala quantitativa inclui os tipos de dados de intervalo e relativo (GAN; MA; WU, 2007). A Figura 2.6 ilustra a classificação dos tipos de atributos.

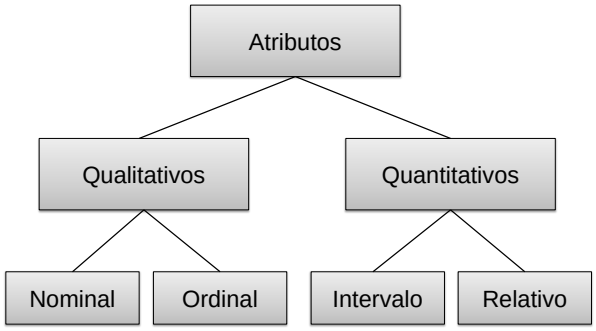

Figura 2.6 – Classificação de atributos do conjunto de dados, adaptado de (GAN; MA; WU, 2007).

Os atributos do tipo nominal possuem valores desordenados, e dessa forma, somente comparações de igualdade são significativas. Isto é, pode-se checar somente se um valor do atributo é o mesmo para duas instâncias. Por exemplo, o atributo "sexo" de uma pessoa, só assume dois possíveis valores nominais: "feminino" ou "masculino". No tipo ordinal, valores de atributos são ordenados, dessa forma, além de comparações de igualdade é possível quantificar a diferença entre dois valores. Por exemplo, no atributo "escolaridade" os valores podem ser ordenados pela qualificação educacional. Nos atributos de intervalo somente as diferenças fazem sentido ao serem comparadas. Por exemplo, o atributo "temperatura" medido em °C, supondo que está 20 °C em um dia e 10 °C no dia seguinte, só é significativo dizer que a temperatura caiu 10 °C e não que está duas vezes mais frio que o dia anterior. No atributo relativo pode-se computar tanto as diferenças como a proporção entre valores. Por exemplo, para o atributo "idade" de uma pessoa, pode-se dizer que alguém com 20 anos de idade é duas vezes mais velho que uma pessoa de 10 anos de idade.

2.1.2 Medida de Similaridade

Como cada grupo é formado por um subconjunto de instâncias mais similares entre si e dissimilares de instâncias de outros grupos, se faz necessário empregar alguma forma de se obter essa medida de similaridade. A obtenção dessa medida é uma importante tarefa no processo de agrupamento. Nesta etapa, podem existir diversas formas de similaridade a se considerar para o agrupamento, como: a proximidade entre duas instâncias, a proximidade entre uma instância e um grupo e a proximidade entre dois grupos. Para cada uma dessas formas é importante que seja definida uma medida de similaridade adequada de acordo com o domínio de dados manipulado (BARBARA, 2000; FACELI et al., 2011).

Para atributos quantitativos, que são representados por valores numéricos (discretos ou contínuos), três medidas de distância baseadas na métrica de *Minkowski* (Equação 2.1) se destacam na literatura, são elas: a distância **Euclidiana**, distância **Manhattan** e a distância **Chebyschev** (*Supremum*). Todas elas são bastante utilizadas para atributos contínuos, porém, se tratando de atributos discretos (por exempo, binários) a distância Manhattan se destaca (BARBARA, 2000; FACELI et al., 2011; HAN; KAMBER; PEI, 2011).

$$\delta(x_i, x_j) = \sqrt[p]{\sum_{l=1}^{d} \left| x_i^l - x_j^l \right|^p} \tag{2.1}$$

A métrica de *Minskowski* (Equação 2.1) calcula a distância entre duas instâncias x_i e x_j pela somatória da diferença do valor de suas d dimensões. Além disso, podem ser escolhidos diferentes valores para p, sendo $1 \leq p < \infty$. Cada valor de p corresponde a uma variação da métrica, sendo que valores menores para p significam mais robustez e menos sensibilidade a ruídos. Para diferentes valores de p, tais medidas de similaridade são definidas da seguinte maneira:

- **Distância Manhattan:** ($p = 1$) Também conhecida como *City-Block*. A Equação 2.2 define o seu cálculo.

$$\delta(x_i, x_j) = \sum_{l=1}^{d} \left| x_i^l - x_j^l \right| \tag{2.2}$$

- **Distância Euclidiana:** ($p = 2$) A medida de distância comumente utilizada para calcular a distância entre duas instâncias. A Equação 2.3 define o seu cálculo.

$$\delta(x_i, x_j) = \sqrt{\sum_{l=1}^{d} \left(x_i^l - x_j^l \right)^2} \tag{2.3}$$

- **Distância Chebyschev:** ($p = \infty$) Calcula a diferença máxima entre os valores das dimensões de duas instâncias. A Equação 2.4 define o seu cálculo.

$$\delta(x_i, x_j) = \max_{1 \leq l \leq d} \left| x_i^l - x_j^l \right| \tag{2.4}$$

O modelo matemático que descreve uma medida de similaridade é chamado de espaço métrico. Esse espaço métrico é um par $< W, \delta >$, em que W é um domínio de dados e δ uma função de distância (KUTZ et al., 2003). No geral, medidas de similaridade devem obedecer certas propriedades. Segundo (HAN; KAMBER; PEI, 2011), considerando as instâncias x_i, x_j e x_k pertencentes a um conjunto de dados X, uma função de distância $\delta(x_i, x_j)$ deve satisfazer as seguintes propriedades:

- Simetria: $\delta(x_i, x_j) = \delta(x_j, x_i)$;

- Não-negatividade: $0 \leq \delta(x_i, x_j) < \infty$;
- Desigualdade triangular: $\delta(x_i, x_j) \leq \delta(x_i, x_k) + \delta(x_k, x_j)$.

Existem ainda diversas medidas de similaridade, abordando os mais diferentes domínios de dados. Uma visão geral sobre outras medidas de similaridade pode ser encontrada em (COLLINS; OKADA, 2012; SLIMANI, 2013; SAAD; KAMARUDIN, 2013).

2.1.3 Agrupamento

Nesta etapa, a tarefa de dividir instâncias em grupos pode ser realizada de dois diferentes modos: agrupamento não-supervisionado e agrupamento semi-supervisionado. Primeiramente, considerando $X = \{x_1, ..., x_n\}$ um conjunto de n instâncias, o agrupamento não-supervisionado tem o objetivo de encontrar alguma estrutura interessante e significativa no conjunto de dados X, sem utilizar nenhum conhecimento prévio dos dados. Geralmente, é formado um particionamento $\Pi = \{C_1, ..., C_k\}$ de k grupos, representados por centróides $\mu_1, ..., \mu_k$, no qual as instâncias são atribuídas a algum grupo C_j de forma que a distância $\delta(x_i, \mu_j)$ seja minimizada, como é o caso do algoritmo *k-means* (melhor detalhado na Seção 2.2).

Já o agrupamento semi-supervisionado pode contar com algum tipo de informação adicional para guiar o processo de atribuição das instâncias aos grupos. De maneira geral, esse tipo de informação adicional pode ser dada de duas formas. A primeira forma é rotular uma pequena porção do conjunto de dados, induzindo assim, bons grupos na fase inicial dos algoritmos. Outra forma seria informar **restrições**, também a uma pequena quantidade dos dados, que influenciam o processo de atribuição dos dados aos grupos (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; BASU; DAVIDSON; WAGSTAFF, 2008).

Dos diversos tipos de restrições existentes, uma forma simples introduzida em (WAGSTAFF; CARDIE, 2000) se tornou a mais utilizada na literatura da área. Essas restrições são inferidas a um par de instâncias dizendo se elas devem ou não estar juntas em um agrupamento. Ou seja, uma restrição do tipo *must-link* indica que duas instâncias devem pertencer a um mesmo grupo, enquanto que uma restrição do tipo *cannot-link* indica que duas instâncias devem pertencer a grupos distintos. Outros tipos de restrições e como incorporá-las ao processo de agrupamento são descritos no Capítulo 3.

Os algoritmos existentes que realizam o agrupamento são bastante variados, e cada um deles pode assumir um critério que impõe algum tipo específico de estrutura ao agrupamento retornado (FACELI et al., 2011). Como os algoritmos possuem diferentes formas estruturais para realizar o agrupamento, eles podem ser classificados pelos métodos utilizados para gerar os agrupamentos. Esses métodos são (HAN; KAMBER; PEI, 2011):

- **Particionamento:** Dado um conjunto de n instâncias, um método de particionamento constrói k partições dos dados, em que cada partição representa um grupo,

sendo $1 < k \ll n$. Algumas características importantes são que, cada instância deve pertencer a um único grupo, e todas as instâncias devem pertencer a algum grupo. Esse método de particionamento é baseado em distância, isto é, instâncias dentro de um grupo estão mais próximas (são mais similares) do que instâncias em outros grupos. Algoritmos tal como *k-means* (MACQUEEN, 1967), COP-*kmeans* (WAGSTAFF et al., 2001), PAM (KAUFMAN; ROUSSEEUW, 1990), CLARA (KAUFMAN; ROUSSEEUW, 1990) e CLARANS (NG; HAN, 1994) são exemplos de algoritmos de particionamento. O trabalho descrito nesta dissertação utiliza a detecção de agrupamentos por meio de particionamento de dados, e por esta razão, é apresentado com mais detalhes na Seção 2.2.

- **Hierárquico:** Esse método cria uma estrutura hierárquica de um dado conjunto de instâncias. Essa estrutura pode ser construída de duas maneiras: aglomerativa ou divisiva. A abordagem aglomerativa é do tipo *bottom-up*, isto é, começa com cada instância compondo um grupo individual e vai iterativamente unindo os grupos mais próximos até que chegue na quantidade de grupos informada. A abordagem divisiva é do tipo *top-down*, isto é, começa com todas as instâncias em um único grupo e vai dividindo em grupos menores sucessivamente até a quantidade de grupos informada ser alcançada ou até que só haja uma instância por grupo. Os algoritmos BIRCH (ZHANG; RAMAKRISHNAN; LIVNY, 1996), CURE (GUHA; RASTOGI; SHIM, 1998), OPTICS (ANKERST et al., 1999) e ROCK (GUHA; RASTOGI; SHIM, 2000) são exemplos de algoritmos hierárquicos.

- **Baseado em Densidade:** Geralmente, métodos baseados em particionamento e distância entre instâncias são bons para deduzir grupos de forma esférica, porém, quando os grupos têm formas arbitrárias esses métodos se mostram ineficientes. Com isso, o método de agrupamento baseado em densidade foi desenvolvido para suprir essa necessidade. A ideia geral é aumentar o tamanho de um determinado grupo até que a densidade de instâncias em um determinado raio exceda algum dado *threshold*. Tal método também é muito usado na filtragem de ruídos e *outliers*. Algoritmos como o DENCLUE (HINNEBURG; KEIM, 1998), DBSCAN (ESTER et al., 1996) e *Wave-cluster* (SHEIKHOLESLAMI; CHATTERJEE; ZHANG, 1998) são alguns dos que usam essa abordagem.

- **Baseado em Grade:** Métodos baseados em grade quantizam o espaço de dados dentro de um número finito de células, formando uma estrutura em grade. Sua principal vantagem está em seu rápido tempo de processamento, sendo ideal para grandes conjuntos de dados, além de se comportar bem com a presença de *outliers*. Os algoritmos CLIQUE (AGRAWAL et al., 1998), MAFIA (NAGESH; GOIL; CHOUDHARY, 2001), STING (WANG; YANG; MUNTZ, 1997) e OptiGrid

(HINNEBURG; KEIM, 1999) se destacam como exemplos que usam a abordagem baseada em grade.

2.1.4 Validação

A etapa de validação surge com o propósito de avaliar e comparar diferentes algoritmos de agrupamento e verificar as estruturas encontradas por eles a fim de identificar algum significado. Segundo (ZAKI; MEIRA, 2014), essa etapa de validação é composta por três partes principais, sendo elas: a avaliação do agrupamento, a estabilidade do agrupamento e a tendência do agrupamento. A avaliação do agrupamento busca avaliar o nível de qualidade ou satisfação do agrupamento. A estabilidade do agrupamento procura entender a sensibilidade do agrupamento resultante a vários parâmetros do algoritmo. Por fim, a tendência do agrupamento avalia a adequação da aplicação do agrupamento, isto é, verifica se os dados possuem alguma estrutura de agrupamento (FACELI et al., 2011; ZAKI; MEIRA, 2014).

Geralmente, a avaliação do agrupamento é baseada em índices estatísticos, que podem quantificar a qualidade de um determinado agrupamento. A escolha de um índice dentre os vários existentes se dá pelo **critério de validação**, que é a estratégia adotada para validar a estrutura de um agrupamento pelo valor estatístico revelado por um índice. Existem três tipos de critério de validação, detalhados a seguir (ZAKI; MEIRA, 2014):

- **Critério Externo:** É uma medida empregada que não é inerente ao conjunto de dados. Esta medida pode estar na forma de conhecimento prévio de um especialista do domínio sobre os grupos. Por exemplo, considerando um conjunto de dados em que seus rótulos já são conhecidos, o critério externo pode medir a correspondência desse conjunto com os grupos obtidos de um processo de agrupamento. Dentre os índices para esse critério estão o índice *Rand*, *Jaccard*, *Fowlkes* & *Mallows*, *Hubert* Normalizado, *Rand* Corrigido e Variação de Informação (HALKIDI; BATISTAKIS; VAZIRGIANNIS, 2002).

- **Critério Interno:** É uma medida de validação derivada do próprio conjunto de dados. Isto é, a partir de informações obtidas do próprio conjunto de dados, tal como uma matriz de similaridade, esse critério pode inferir o quanto os grupos de um processo de agrupamento representam a estrutura real dos dados. O índice *Gap* (TIBSHIRANI; GUENTHER; HASTIE., 2001) é um exemplo usado para esse tipo de critério.

- **Critério Relativo:** É uma medida com o objetivo de comparar diretamente diferentes agrupamentos obtidos por diferentes configurações de parâmetros para um mesmo algoritmo. Pode ser utilizado na comparação de diferentes algoritmos e na decisão da melhor configuração de parâmetros de um algoritmo para o agrupamento.

Índices como Variância Intracluster, Conectividade, *Dunn* e Silhueta são exemplos para esse critério (VENDRAMIN; CAMPELLO; HRUSCHKA, 2010).

Particularmente no trabalho descrito nesta dissertação, o índice *Rand* foi utilizado para validar os particionamentos gerados pelo método de agrupamento de dados desenvolvido. Dentre as razões para essa escolha, foi observado que se trata de um índice de validação bastante utilizado na literatura de agrupamento de dados. Além disso, os rótulos dos dados das bases utilizadas durante os experimentos já são conhecidos, o que reforça ainda mais o uso de tal índice. Dessa forma, o índice *Rand* é melhor detalhado logo a seguir nesta seção.

2.1.4.1 Índice *Rand*

No trabalho descrito em (RAND, 1971) é apresentado um índice de validação que tem o objetivo de comparar dois agrupamentos arbitrários baseado em como os pares de instâncias são agrupados. A partir de dois agrupamentos de um mesmo conjunto de dados, para qualquer par de instâncias é verificada duas situações: primeiramente verifica se as duas instâncias foram atribuídas ao mesmo grupo em cada um dos agrupamentos ou se foram atribuídas a grupos distintos em ambos os agrupamentos; logo após, verifica se as duas instâncias foram atribuídas ao mesmo grupo em um agrupamento e em grupos diferentes no outro agrupamento. Esse índice será mais detalhado logo a seguir.

Considerando um conjunto de dados $X = \{x_1, x_2, ..., x_n\}$ de n instâncias, seja $\Pi' = \{C'_1, C'_2, ..., C'_k\}$ e $\Pi'' = \{C''_1, C''_2, ..., C''_k\}$ dois distintos particionamentos do conjunto X. Com isso, o índice *Rand* é definido pela a Equação 2.5 (GAN; MA; WU, 2007), como segue.

$$rand\,(\Pi', \Pi'') = \frac{1}{\binom{n}{2}} \sum_{1 \leq i < j \leq n} \gamma_{ij} \tag{2.5}$$

sendo que

$$\gamma_{ij} = \begin{cases} 1 & \text{se } \exists l, t \text{ tal que } x_i \in C'_l \cap C''_t \text{ e } x_j \in C'_l \cap C''_t \\ 1 & \text{se } \exists l, t \text{ tal que } x_i \in C'_l \cap C''_t \text{ e } x_j \notin C'_l \cup C''_t \\ 0 & \text{caso contrário} \end{cases}$$

Seja n_{ij} o número de instâncias que estão simultaneamente no i-ésimo grupo de Π' e o j-ésimo grupo de Π'', ou seja, $n_{ij} = |C_i, C_j|$. Desta forma, a função $rand\,(\Pi', \Pi'')$ pode ser simplificada como mostra a Equação 2.6, ilustrada a seguir.

$$rand\,(\Pi', \Pi'') = 1 + \frac{1}{\binom{n}{2}} \sum_{i=1}^{k} \sum_{j=1}^{k} n_{ij}^2 - \frac{1}{2\binom{n}{2}} \left[\sum_{i=1}^{k} \left(\sum_{j=1}^{k} n_{ij} \right)^2 + \sum_{j=1}^{k} \left(\sum_{i=1}^{k} n_{ij} \right)^2 \right] \tag{2.6}$$

Nesta função, pode-se perceber que $rand\,(\Pi', \Pi'')$ é uma medida de similaridade com valor variando no intervalo [0,1]. Quando o valor retornado para dois agrupamentos for igual a zero (0), significa que eles não tem nenhuma similaridade. No momento em que o valor retornado for igual a um (1), significa que os dois agrupamentos são idênticos.

2.1.4.2 Validação Estatística

Em um ambiente de experimentação de diversos algoritmos em diversas bases de dados, o resultado ou a média de resultados gerados por um determinado índice de validação pode não ser suficiente para definir diferenças de desempenho entre os algoritmos. Vários pesquisadores adotam diferentes técnicas estatísticas ou de senso comum para decidir se as diferenças entre os algoritmos são reais ou aleatórias. Para esse propósito, em muitos casos, é aplicado o teste de hipóteses para avaliar os resultados de metodologias de avaliação (DEMSAR, 2006).

Geralmente, o teste de hipóteses possui um conjunto de passos predeterminados que servem como guia durante a execução do mesmo, como segue. A primeira etapa é formular as hipóteses, nessa fase é necessário estabelecer as hipóteses nula (H_0) e alternativa (H_A). Normalmente, em experimentos de comparação de algoritmo a hipótese nula H_0 afirma que o desempenho dos algoritmos testados são equivalentes. Após formular a hipótese nula é conveniente formular uma hipótese alternativa H_A, a qual pode ser aceita desde que a hipótese nula seja rejeitada. O próximo passo é determinar um nível de significância, ou nível de confiança para o teste, isto é, fixar a probabilidade de se cometer o erro do tipo I[1]. Ao fixar a probabilidade de ocorrer o erro do tipo I, é possível determinar o valor crítico considerado (dado em tabela), o qual irá separar a região de rejeição da região de aceitação. Após o cálculo da estatística é possível saber se o resultado está na região crítica, e então rejeitar H_0, ou caso contrário, aceitar H_0.

Para decidir entre a aceitação ou rejeição de uma determinada hipótese é necessário calcular a estatística dos resultados gerados por metodologias de avaliação que comparam diferentes algoritmos em diversos conjuntos de dados. Nesse cenário, o teste paramétrico ANOVA e o teste não-paramétrico de *Friedman* se mostram como solução para esse problema em questão. Entretanto, o teste ANOVA é um teste paramétrico, no qual exige algumas características aos conjuntos de dados relacionados aos experimentos. Dentre essas características, um teste paramétrico pressupõem que os dados obedecem uma distribuição normal, possuem uma variância homogênea e que estejam contidos em um intervalo contínuo. Por outro lado, um teste não-paramétrico requer menos exigências a distribuição dos dados e é mais utilizado em teste de hipóteses. Assim, o teste de *Friedman* aparece como uma solução adequada para realizar o teste estatístico quando não se possui grandes exigências aos vários conjuntos dados, fornecendo assim, um teste de

[1] Em um teste de hipóteses, um erro do tipo I consiste em rejeitar a hipótese nula mesmo quando é verdadeira.

significância das diferenças entre os vários algoritmos. O teste de *Friedman* é apresentado a seguir.

Teste de *Friedman*

O teste de *Friedman*, inicialmente descrito em (FRIEDMAN, 1937), é um teste não-paramétrico que ranqueia os algoritmos para cada conjunto de dados separadamente. Por exemplo, um algoritmo A teve melhor desempenho que um algoritmo B para um determinado conjunto de dados X, neste caso, o algoritmo A será ranqueado em primeiro lugar e o algoritmo B em segundo lugar. Em caso de empates, ou seja, dois algoritmos com o mesmo desempenho, será atribuído o *rank* médio para esses dois algoritmos. Por exemplo, se os algoritmos A e B estão empatados em seus valores de desempenho e como não há uma forma de escolher qual dos dois algoritmos será ranqueado em primeiro lugar, então será atribuído a soma dos *ranks* ocupados pelos algoritmos (1 e 2) dividido pelo total de algoritmos empatados. Isso resulta em *rank* = 1,5 para ambos os algoritmos.

Considerando ρ_i^j o *rank* do j-ésimo algoritmo, de um total de t algoritmos, no i-ésimo conjunto de dados em N existentes. O teste de *Friedman* compara o *rank* médio do algoritmo como, $\bar{P}_j = \frac{1}{N}\sum_i \rho_i$, sendo $\bar{P}_j$ o valor de *rank* médio de um algoritmo j para os N conjuntos de dados. Com a hipótese nula (H_0), a qual afirma que todos os algoritmos são equivalentes, a estatística de *Friedman* é computada de acordo com a Equação 2.7 (DEMSAR, 2006), como segue.

$$\chi_F^2 = \frac{12N}{t\,(t+1)}\left[\sum_{j=1}^{t}\bar{P}_j^2 - \frac{t\,(t+1)^2}{4}\right] \tag{2.7}$$

A estatística de *Friedman* é distribuída de acordo com χ_F^2, considerando $t-1$ graus de liberdade. Essa estatística é utilizada quando N e t são suficientemente grandes, como $N > 10$ e $t > 5$, para menores números de algoritmos e de conjunto de dados a estatística utilizada deve ser a apresentada na Equação 2.8 (ZAR, 2007), a qual utiliza a soma dos *ranks* $P_j = \sum_i \rho_i$ em vez do valor de *rank* médio $\bar{P}$, como ilustrada a seguir:

$$\chi_r^2 = \frac{12}{Nt\,(t+1)}\left[\sum_{j=1}^{t}P_j^2 - 3N\,(t+1)\right] \tag{2.8}$$

Entretanto, os autores do trabalho descrito em (IMAN; DAVENPORT, 1980) mostraram que a estatística de *Friedman* é indesejavelmente conservadora e necessitaria ser derivada em uma melhor estatística. Essa nova estatística, mostrada na Equação 2.9, é distribuída de acordo com a F-*distribution*, considerando $t-1$ e $(t-1)\,(N-1)$ graus de liberdade.

$$F_F = \frac{(N-1)\,\chi_F^2}{N\,(t-1) - \chi_F^2} \tag{2.9}$$

A partir desses valores de estatística, como o F_F ou χ^2_F, um valor crítico deve ser consultado de acordo com os respectivos graus de liberdade. As tabelas com os valores críticos podem ser encontradas em (ZAR, 2007), veja Apêndice A.1. A hipótese nula (H_0) será rejeitada se o valor da estatística F_F ou χ^2_F for maior que o valor crítico, confirmando assim, que existe diferença significativa entre os t algoritmos testados. Porém, o teste de *Friedman* tem como a sua principal funcionalidade dizer apenas se existe uma diferença significativa de desempenho entre os algoritmos. Para verificar qual algoritmo é superior aos demais, um teste *post-hoc* deve ser realizado, podendo ser feito de duas formas: comparando todos os algoritmos com todos os outros (*Nemenyi*); ou fixando um algoritmo específico e comparando com os algoritmos restantes (*Bonferroni-Dunn*). Diante da situação descrita nesta dissertação, onde o trabalho propôs um único algoritmo, necessita-se apenas comparar esse único algoritmo com os outros algoritmos da literatura. O teste necessário para realizar essa tarefa é descrito a seguir.

Teste *post-hoc Bonferroni-Dunn*

Quando a hipótese nula (H_0) é rejeitada, o teste de *Bonferroni-Dunn*, apresentado em (DUNN; DUNN, 1961), é utilizado para comparar um determinado algoritmo de controle com os demais algoritmos que estão sendo testados. Para cada par de algoritmos, o desempenho será significativamente diferente se a média dos *ranks* correspondentes diferem por pelo menos a diferença crítica (CD, do inglês *critical difference*). A diferença crítica é dada por meio da Equação 2.10, apresentada a seguir.

$$CD = q_\alpha \sqrt{\frac{t\,(t+1)}{6N}} \tag{2.10}$$

Nesta equação, q_α é o valor crítico baseado na *Studentized Range Statistic* dividida por $\sqrt{2}$, e α é o nível de confiança desejado para o teste. A tabela com esses valores críticos pode ser encontrada em (DEMSAR, 2006), veja Apêndice A.2. Após calculada a diferença crítica, o próximo passo será calcular a diferença de desempenho de dois determinados algoritmos. Essa diferença de desempenho é calculada pela Equação 2.11, como segue.

$$z = \frac{\left(\bar{P}_i - \bar{P}_j\right)}{\sqrt{\frac{t(t+1)}{6N}}} \tag{2.11}$$

Para dois valores de *rank* médio $\bar{P}_i$ e $\bar{P}_j$ de dois respectivos algoritmos, se o valor z for maior que a diferença crítica CD significa que um algoritmo é estatisticamente superior ao outro. Neste caso, o algoritmo de menor *rank* médio, ou seja, o algoritmo que teve os melhores ranqueamentos nos N conjuntos de dados, será considerado superior.

2.2 Detecção de Agrupamentos por Particionamento

A etapa de agrupamento é a parte central de todo o processo, a qual consiste em agrupar os dados mais similares de um conjunto de dados. Nesta etapa, mais de um algoritmo pode ser aplicado aos dados para identificar as possíveis estruturas de grupos existentes. Cada algoritmo busca por uma única estrutura que se ajuste da melhor forma aos dados, e em alguns casos, pode haver mais de uma estrutura, cada qual representando uma visão diferente desses dados.

Em (FACELI et al., 2011), os autores definem um conceito para **partições** embasando-se na teoria de conjuntos, no qual uma partição dos dados é a divisão de um conjunto de instâncias em subconjuntos menores, aqui intitulados como grupos. A fim de formalizar esse conceito, considere um conjunto de dados $X = \{x_1, ..., x_n\}$, uma partição de X em k grupos é definida como $\Pi = \{C_1, ..., C_k\}$ com $k \ll n$, tal que Π possua as seguintes propriedades:

- Todos os grupos devem conter pelo menos uma instância;

$$C_j \neq \emptyset, \quad j = 1, ..., k$$

- Todas as instâncias devem pertencer a algum grupo;

$$X = \bigcup_{j=1}^{k} C_j$$

- Cada instância deve pertencer exclusivamente a um único grupo.

$$C_i \cap C_j = \emptyset, \quad i, j = 1, ..., k; i \neq j$$

Um dos algoritmos por particionamento mais conhecidos é o algoritmo *k-means* (MACQUEEN, 1967). O algoritmo *k-means* utiliza um método baseado em centróide, isto é, a cada iteração o algoritmo atribui cada instância x_i do conjunto de dados X a um grupo C_j cujo centróide μ_j se encontra mais próximo. Cada grupo possui um centróide representante, dado pela média aritmética de todas as instâncias pertencentes a esse grupo, como mostrado na Equação 2.12, em que $|C_j|$ é a cardinalidade de um grupo C_j.

$$\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i \tag{2.12}$$

O Algoritmo 1 apresenta todos os passos do algoritmo *k-means*. Primeiramente, o *k-means* recebe como entrada um conjunto de dados X e um número k de grupos a ser definido no agrupamento. Então, o algoritmo seleciona aleatoriamente k instâncias do conjunto para representarem os centróides iniciais. O processo principal se resume em atribuir as instâncias aos grupos com centróides mais próximos e atualizar o centróide de

Algoritmo 1 - *k-means*

Entrada: Conjunto de dados X, Número de grupos k
Saída: Partições $\Pi = \{C_1, ..., C_k\}$
  Seleciona aleatoriamente $\mu_1, ..., \mu_k$ como centróides iniciais
  repita
    para cada instância $x_i \in X$ **faça**
      Atribui x_i ao grupo C_j mais próximo
    fim para
    para cada grupo C_i **faça**
      Atualiza o centróide μ_i pela média de todas as instâncias x_j atribuídas a ele
    fim para
  até convergir
  retorna $\Pi = \{C_1, ..., C_k\}$

cada grupo conforme a Equação 2.12. Esse processo se repete até convergir, ou seja, até que os centróides se estabilizem por uma determinada função objetivo. Como resultado são obtidas as k partições do conjunto de dados.

Geralmente, algoritmos de agrupamento são baseados em tentar minimizar ou maximizar uma função objetivo global. Com isso, o problema de agrupamento passa a ser um problema de otimização, o qual pode ser resolvido, de maneira geral, quantificando todas as formas possíveis de dividir os dados em grupos e avaliando o potencial desses grupos por uma dada função objetivo (BARBARA, 2000). Por exemplo, o algoritmo *k-means* tecnicamente só pode parar se:

$$\sum_{i=1}^{k} \left\| \mu_i^t - \mu_i^{t-1} \right\|^2 \leq \epsilon$$

para um dado limite $\epsilon > 0$, em que t representa a atual iteração. Isto é, só deve parar quando a diferença do valor de um centróide μ_i entre o tempo t e $t-1$ for menor ou igual a um limite ϵ, geralmente um valor bem próximo de zero.

2.3 Múltiplos Protótipos

A estratégia de múltiplos protótipos aparece, na literatura da área, com propostas de algoritmos de detecção de agrupamentos, na sua maioria, particionais e hierárquicos, que utilizam mais de um representante para cada grupo no processo de agrupamento. Geralmente, esses representantes extras podem ser gerados por um processo de seleção de instâncias aleatórias ou por meio de alguma informação adicional do próprio conjunto de dados que possa ser usada para defini-los. Contudo, a utilização de múltiplos protótipos pode ser divida em duas principais abordagens, da seguinte forma: a primeira pode considerar um número p de protótipos, com $p > k$, dividindo o conjunto de dados em p grupos, e em seguida, os grupos são unidos a fim de obter os k grupos inicialmente informados; a segunda abordagem considera os k grupos com diversos representantes em cada um de-

les, sendo utilizados para calcular a similaridade de uma instância para um determinado grupo ou entre dois grupos distintos. A estratégia, de utilizar múltiplos protótipos, tem como principal finalidade representar mais facilmente grupos de formas arbitrárias, superando assim, a dificuldade de algoritmos que utilizam apenas um representante por grupo. Alguns algoritmos que utilizam a abordagem de múltiplos protótipos para a detecção de agrupamentos são apresentados a seguir.

Seguindo a primeira abordagem, o algoritmo descrito em (LIU; JIANG; KOT, 2009) utiliza uma estratégia de seleção aleatória para recrutar novos representantes ao processo de agrupamento. Primeiramente, o algoritmo seleciona aleatoriamente p instâncias como representantes, sendo $p > k$, e divide o conjunto de dados em p partições. Em seguida, para cada grupo, se o erro quadrático é maior que um dado limite, um novo protótipo é adicionado a esse grupo. Enquanto que, se um grupo tem um número de instâncias relativamente baixo, seu protótipo é excluído. A cada nova inserção ou remoção de representantes, o algoritmo particiona os dados na quantidade de protótipos atual. Por fim, os p representantes são agrupados em k grupos e verifica se a densidade na borda de um par de grupos é relativamente alta ou se a quantidade de instâncias também é relativamente alta. Nesses casos, um novo protótipo é adicionado na borda desse par de grupos e o algoritmo volta a particionar os grupos em p partições. Caso a densidade seja relativamente baixa e a quantidade de instâncias não seja tão elevada, o algoritmo finaliza retornando as k partições.

No entanto, o trabalho apresentado em (LUO et al., 2010) aborda a seleção aleatória do trabalho descrito em (LIU; JIANG; KOT, 2009) como um problema e propõe uma solução baseada em uma representação dos dados usando uma MST (*Minimal Spanning Tree*). Nesse trabalho, os múltiplos protótipos são gerados com base no grau de vizinhança de cada nó. Isto é, qualquer nó que tenha uma quantidade de vizinhos maior ou igual a um dado limite informado se torna um protótipo. Para esse caso particular, entenda nó como uma instância do conjunto de dados. Após a definição dos múltiplos protótipos, o algoritmo atribuí cada instância ao representante mais próximo. Como o número de protótipos geralmente será maior que a quantidade k de grupos, uma etapa realiza a união dos protótipos mais próximos com base em um critério. Esse algoritmo pode ter várias iterações até alcançar a quantidade k de grupos desejados. Mais detalhes podem ser encontrados em (LUO et al., 2010).

O algoritmo CURE (GUHA; RASTOGI; SHIM, 1998) é um dos algoritmos hierárquicos clássicos em detecção de agrupamentos que utiliza a abordagem de múltiplos representantes para cada grupo. Se trata de um algoritmo aglomerativo, em que, a cada etapa, dois grupos mais próximos são unidos a fim de formar um único grupo, até que a quantidade de grupos desejada seja alcançada. Os múltiplos representantes, então, são gerados por meio da seleção de instâncias bem distribuídas na região de um determinado grupo, como segue: para cada grupo, seleciona uma instância que está mais distante do

centróide, em seguida seleciona uma instância mais distante da instância selecionada anteriormente, nos passos seguintes são selecionadas instâncias mais distantes das que já foram selecionadas até que uma quantidade máxima seja alcançada. Esses representantes são utilizados para medir a distância entre dois grupos distintos, tendo em vista que realizar o cálculo para todas as instâncias de dois grupos teria um custo computacional bastante elevado ao longo de todo o processo.

Uma abordagem mais parecida com o trabalho descrito nesta dissertação é apresentada no trabalho em (HUANG; CHENG; ZHAO, 2008). Nesse caso, o algoritmo considera informações de semi-supervisão entre pares de instâncias para gerar um representante a mais para cada grupo. Isto é, cada grupo conta com dois representantes, um com base na média de instâncias representadas por esse grupo e outro com base em informações de semi-supervisão. Porém, esse algoritmo poderia ter melhores resultados se aproveitasse as informações de semi-supervisão para gerar não apenas um, mas vários representantes para cada grupo. Essa deficiência é explorada pelo trabalho descrito nesta dissertação, utilizando vários protótipos para representar de forma mais eficaz as estruturas complexas encontradas em grupos de formas arbitrárias. Contudo, por ser uma abordagem de semi-supervisão, possui muitas características em comum com o algoritmo abordado nesta dissertação, e por esse motivo, será melhor detalhado no Capítulo 3.

2.3.1 Medida de Similaridade Agregada

Para as estratégias que consideram todos os múltiplos protótipos de um grupo no cálculo da similaridade, o trabalho apresentado em (RAZENTE et al., 2008) define uma medida de similaridade que pode ser facilmente adaptada para computar a similaridade em um processo de agrupamento com múltiplos representantes, denominada **similaridade agregada**. O cálculo de múltiplos representantes considera um conjunto de protótipos Q para cada grupo, de forma que a distância $\delta(x_i, x_q)$ entre cada protótipo $x_q \in Q$ e uma instância $x_i \in X$ deve ser calculada por uma função de similaridade agregada δ_g.

Existem diversas formas de se definir a função de similaridade agregada. Na adaptação do trabalho descrito em (RAZENTE et al., 2008; RAZENTE, 2009), a função δ_g entre um conjunto de protótipos Q e uma instância $x_i \in X$ é definida na Equação 2.13, como segue:

$$\delta_g(Q, x_i) = \sqrt[g]{\sum_{x_q \in Q} [\delta(x_q, x_i)^g \cdot w_q]} \tag{2.13}$$

sendo $\delta()$ uma função de similaridade, w_q um peso correspondente ao protótipo representante x_q e $g \in \mathbb{R}^+$ um valor real diferente de zero denominado **fator de agregação**.

Variações de valor do fator de agregação podem modificar a área de abrangência de um protótipo. A Figura 2.7 mostra o efeito dessas variações no fator de agregação g em um espaço bidimensional, considerando quatro protótipos distintos e o uso da distância

Euclidiana. Na figura, as linhas correspondem à posições geométricas em que a Equação 2.13 resulta no valor relativo aos protótipos $x_q \in Q$. Nota-se que para valores de $g < 1$ podem ser obtidas regiões disjuntas.

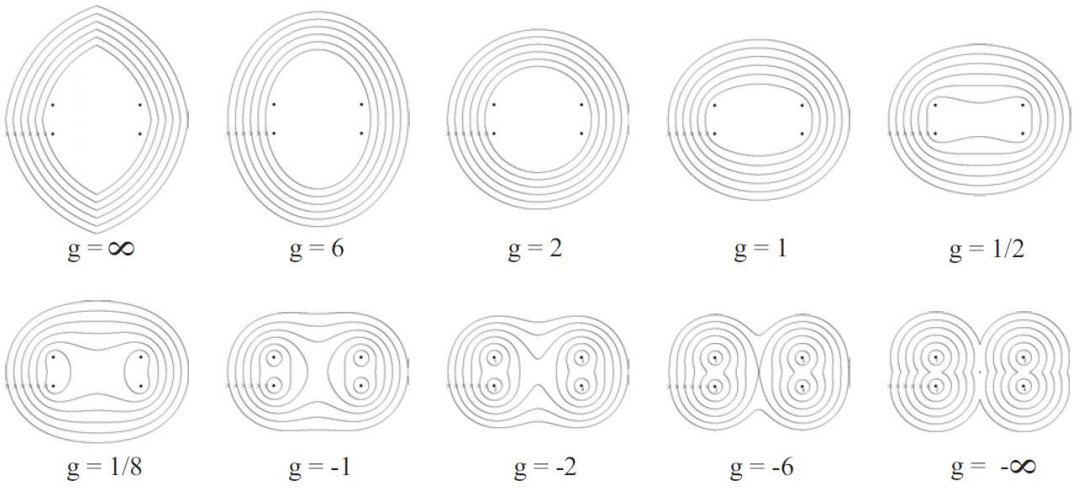

Figura 2.7 – Efeito do fator de agregação g no espaço euclidiano (RAZENTE, 2009).

A função de similaridade agregada ainda pode ser minimizada em duas funções de características especiais. Sendo uma minimização da similaridade agregada máxima, considerando o fator $g = \infty$, e também, uma minimização da similaridade agregada mínima, considerando o fator $g = -\infty$. Ambas minimizações são casos especiais da Equação 2.13 e estão ilustradas nas respectivas Equações 2.14 e 2.15, como segue.

$$\delta_{g=\infty}(Q, x_i) = \lim_{g\to\infty} \sqrt[g]{\sum_{x_q \in Q} \delta(x_q, x_i)^g} = \max(\delta(x_q, x_i)), \quad \forall x_q \in Q \tag{2.14}$$

$$\delta_{g=-\infty}(Q, x_i) = \lim_{g\to-\infty} \sqrt[g]{\sum_{x_q \in Q} \delta(x_q, x_i)^g} = \min(\delta(x_q, x_i)), \quad \forall x_q \in Q \tag{2.15}$$

Nesses casos especiais da função de similaridade agregada, as distâncias retornadas quando calculadas entre um conjunto de representantes Q e uma instância x_i possuem comportamentos totalmente opostos. A Equação 2.14, retornará a distância máxima de um conjunto de representantes para uma instância, ou seja, ela retorna a distância do representante mais distante de x_i. Já a Equação 2.15, retornará a distância mínima de um conjunto de representantes para uma instância, ou seja, irá retornar a distância do representante mais próximo de x_i. Esse segundo caso, a função de similaridade agregada mínima, também pode ser encontrada no trabalho descrito em (GUHA; RASTOGI; SHIM, 1998), o qual comparava a distância entre dois grupos pela distância mínima de seus representantes.

2.4 Considerações Finais

Este capítulo apresentou todos os detalhes referentes ao processo de detecção de agrupamentos de dados, enfatizando todas as etapas que regem o processo como um todo. O agrupamento de dados por particionamento obteve maior destaque e foram apresentadas algumas propriedades relacionadas a essa abordagem. Outra forma de se realizar agrupamento é por meio de uma técnica semi-supervisionada, em que informações adicionais sobre o conjunto de dados podem auxiliar o processo de detecção de agrupamento.

Também foram apresentadas algumas técnicas de utilização de múltiplos protótipos em detecção de agrupamentos, exemplificando com algoritmos de abordagens por particionamento e hierárquicas. Essas estratégias de múltiplos representantes têm sido bastante utilizadas com o objetivo de detectar formas diversas em agrupamento de conjuntos de dados. Entretanto, sua utilização em abordagens de agrupamentos por particionamento não tem tido todo o seu potencial explorado até o momento. Isso motiva a criar mais formas de gerar múltiplos protótipos e formas de utilização dentro de um processo de particionamento de dados.

Por fim, foi apresentada a medida de similaridade agregada, que pode ser utilizada para medir a similaridade entre instâncias e representantes de grupos no processo de agrupamento. No próximo capítulo são apresentadas técnicas de semi-supervisão e formas de incorporação da informação de semi-supervisão ao processo de detecção de agrupamentos, bem como, uma metodologia de avaliação específica para agrupamentos semi-supervisionado de dados.

Capítulo 3

Semi-Supervisão

A detecção de agrupamentos é uma importante ferramenta para mineração de dados, uma vez que ela pode identificar padrões e tendências sem qualquer informação além da distribuição espacial dos dados. Entretanto, em muitos casos têm-se acesso às informações adicionais ou conhecimento de domínio sobre os dados, que indicam qual estrutura se busca para um conjunto de dados. Essas informações adicionais ocorrem, na maioria dos casos, no nível da instância, tal como, rótulos a um subconjunto de instâncias, informação complementar sobre a verdadeira similaridade entre pares de instâncias ou preferências de usuários sobre como as instâncias deveriam ser agrupadas. Também pode ocorrer em um nível mais elevado, podendo codificar o conhecimento sobre os próprios grupos, tal como, sua posição, identidade, tamanho máximo ou minimo e distribuição (BASU; DAVIDSON; WAGSTAFF, 2008).

O agrupamento semi-supervisionado surgiu a partir da necessidade de encontrar formas de utilizar esta informação quando ela eventualmente está disponível. Isso caracteriza um aprendizado semi-supervisionado, que é um meio termo entre o aprendizado supervisionado e o não-supervisionado. Nesse contexto, além dos dados não rotulados, os algoritmos recebem algum tipo de informação de supervisão, mas não necessariamente para todas as instâncias. Mesmo sendo possível que um algoritmo de agrupamento completamente não supervisionado encontre naturalmente um particionamento que seja consistente como o conhecimento de domínio, os casos mais interessantes são aqueles em que o conhecimento de domínio sugere um agrupamento não convencional para os dados. Isso têm motivado pesquisadores a explorar métodos que impõem propriedades desejáveis ao agrupamento (CHAPELLE; SCHÖLKOPF; ZIEN, 2006; BASU; DAVIDSON; WAGSTAFF, 2008).

Este capítulo tem como objetivo abordar o agrupamento semi-supervisionado com restrições, considerando o problema de particionar um conjunto de instâncias em um especificado número de grupos quando uma quantidade limitada de supervisão é fornecida na forma de restrições em um nível de, por exemplo, instâncias, grupos ou atributos. Enquanto a detecção de agrupamentos é tradicionalmente considerada uma forma de apren-

dizado não supervisionado, a inclusão de restrições a torna uma tarefa de aprendizado semi-supervisionado, onde o desempenho dos algoritmos não supervisionados podem ser melhorados utilizando essa informação como uma espécie de dados de treinamento. Contudo, essas restrições devem refletir os objetivos de uma determinada tarefa, por exemplo, usar o agrupamento para determinar uma partição dos dados que refletem as estruturas das classes inerentes, ou seja, criar grupos que se assemelhem ao máximo à estrutura real ou desejada para os dados.

O capítulo está organizado da seguinte forma. A Seção 3.1 aborda alguns dos variados tipos de restrições existentes na literatura. A Seção 3.2 apresenta formas de incorporar as restrições ao processo de detecção de agrupamentos. Na Seção 3.3 é apresentada uma metodologia de avaliação específica para algoritmos de agrupamento semi-supervisionados. Por fim, a Seção 3.4 traz as considerações finais do capítulo.

3.1 Tipos de Restrição

Existem diversos tipos de restrições descritos na literatura, englobando diferentes abordagens. Alguns desses tipos de restrições divergem na forma em que representam o conhecimento, e também, são bastante eficientes no processo de detecção de agrupamentos, no qual se tem pouca informação sobre sua estrutura e o seu domínio. Essas restrições podem atuar tanto em domínios específicos como abranger uma forma mais genérica, podendo ser usadas em um amplo contexto. As subseções seguintes descrevem alguns dos tipos de restrições encontrados na literatura.

3.1.1 Restrições em Nível de Instância

Foi proposto em (WAGSTAFF; CARDIE, 2000) uma forma de restrição que trabalha no nível da instância, em que informações sobre pares de instâncias desempenham um papel fundamental no processo de agrupamento. Essas restrições são divididas em dois tipos, e são chamadas de restrições *must-link* e *cannot-link*. Restrições do tipo *must-link* definem que um par de instâncias deve pertencer ao mesmo grupo, enquanto restrições do tipo *cannot-link* definem que um par de instâncias não pode pertencer ao mesmo grupo. Ou seja, considerando um conjunto de dados $X = \{x_1, ..., x_n\}$, uma restrição *must-link* é definida como: $r_{ml}(x_i, x_j)$, em que x_i deve pertencer ao mesmo grupo de x_j, sendo $i \neq j$. Já uma restrição *cannot-link* é definida como: $r_{cl}(x_i, x_j)$, em que x_i não deve pertencer ao mesmo grupo de x_j, considerando da mesma forma $i \neq j$. A Figura 3.8 mostra um conjunto de instâncias em que cada forma geométrica representa um rótulo de um grupo para o conjunto, pode-se notar a influência das restrições *must-link* e *cannot-link* para a definição final dos grupos no processo.

Esse tipo de restrição expressa bem a informação sobre a estrutura dos dados, permitindo assim, que o algoritmo possa fazer escolhas mais acuradas quando atribui instâncias

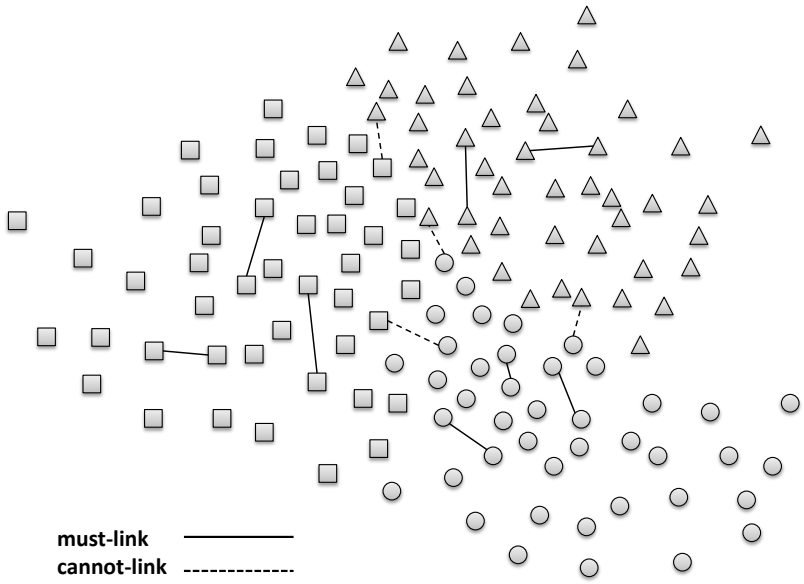

Figura 3.8 – Exemplo de restrições *must-link* e *cannot-link*.

a determinados grupos durante o processo. Em (WAGSTAFF; CARDIE, 2000) os agrupamentos tiveram resultados significativos em diferentes bases de dados de diferentes domínios (numérico, nominal e textual), melhorando a acurácia e diminuindo o tempo de execução. Nesse mesmo trabalho, resultados interessantes também foram obtidos ao aplicar somente restrições *must-link*, ou somente restrições *cannot-link*.

As restrições *must-link* e *cannot-link*, apesar de serem aparentemente simples, compartilham propriedades interessantes. Restrições do tipo *must-link*, por exemplo, possuem uma relação de equivalência e, portanto, são simétricas, reflexivas e transitivas. Isto significa que $r_{ml}(x_i, x_j)$ e $r_{ml}(x_j, x_k)$ implicam em $r_{ml}(x_i, x_k)$ tal que x_i, x_j e x_k formam uma componente conexa, ou seja, cada instância está conectada a outra explicitamente ou implicitamente por uma restrição *must-link*. De maneira formal, considere χ_p e χ_t componentes conexas (subconjuntos de instâncias interconectadas por restrições *must-link*) e que x_i e x_j são instâncias em χ_p e χ_t, respectivamente. Assim, $r_{ml}(x_i, x_j)$, $x_i \in \chi_p$, $x_j \in \chi_t \Rightarrow r_{ml}(x_k, x_l)$, $\forall x_k, x_l : x_k \in \chi_p$, $x_l \in \chi_t$. Semelhantemente, as componentes conexas de restrições *must-link* podem implicar em restrições do tipo *cannot-link*, considerando χ_p e χ_t uma componente conexa e que x_i e x_j são instâncias em χ_p e χ_t, respectivamente. Então $r_{cl}(x_i, x_j)$, $x_i \in \chi_p$, $x_j \in \chi_t \Rightarrow r_{cl}(x_k, x_l)$, $\forall x_k, x_l : x_k \in \chi_p$, $x_l \in \chi_t$ (DAVIDSON; BASU, 2007).

Apesar dos bons resultados de acurácia observados nos trabalhos correlatos da literatura científica, a utilização de pares de restrições *must-link* e *cannot-link* nem sempre fornecem uma orientação satisfatória para o processo de agrupamento. Como por exemplo, em (JIANG et al., 2013) é considerado um problema, ilustrado na Figura 3.9, onde existem 16 instâncias e o objetivo é particionar todas as instâncias em somente dois grupos

de forma que a proximidade dos centróides em relação aos dados de um mesmo grupo seja minimizada. Sob essas especificações só existem duas "partições ótimas", como mostram as Figuras 3.9(a) e 3.9(b). Ambas as figuras apresentam conjuntos diferentes de restrições *must-link*, por linhas sólidas, entre as instâncias. Entretanto, se todas as restrições *must-link* forem satisfeitas simultaneamente, como mostra a Figura 3.9(c), a partição ótima desejada não poderá ser alcançada.

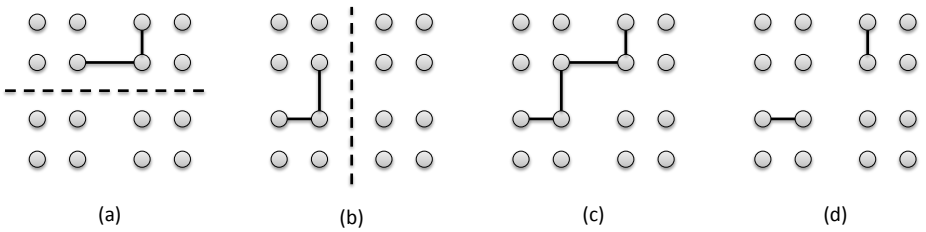

Figura 3.9 – Problema em encontrar "partições ótimas" no conjunto de instâncias de dados, adaptado de (JIANG et al., 2013).

Para abordar esse problema, em (JIANG et al., 2013) foi introduzido um novo conceito para os pares de restrições em nível de instância, que são chamadas de pares de restrições elite, isto é, *elite must-link* (em) e *elite cannot-link* (ec). De forma que, uma restrição $r_{em}(x_i, x_j)$ indica que as instâncias x_i e x_j devem aparecer juntas em todas as partições ótimas possíveis. Semelhantemente, uma restrição $r_{ec}(x_i, x_j)$ exige que as instâncias x_i e x_j sejam atribuídas a grupos distintos em todas as partições ótimas, lembrando que $i \neq j$. Voltando ao exemplo anterior, a Figura 3.9(d) mostra o conjunto de restrições *elite must-link* do conjunto de restrições *must-link* que satisfaz a busca por uma partição ótima. Pode-se observar que a maior dificuldade com essa abordagem é definir restrições que produzam uma partição ótima satisfatória, considerando que diferentes conjuntos de dados nem sempre terão a mesma disposição para as instâncias, e que em casos de conjuntos de dados muito grandes essa partição ótima se torna ainda mais difícil de ser encontrada devido ao grande número de possibilidades.

Em outro trabalho, apresentado em (LAI et al., 2014), o autor considera que o uso de detecção de agrupamentos em um contexto de recuperação de imagens por conteúdo de alta dimensionalidade gera melhores resultados, quando comparados com tradicionais métodos de indexação. Nesse trabalho, o usuário pode interativamente inferir sua percepção sobre o agrupamento com realimentações positivas e negativas, relativas a uma imagem que deveria ou não deveria estar em um atual grupo. A partir disso, restrições do tipo *must-link* e *cannot-link* são deduzidas por meio dessas realimentações, e inseridas iterativamente ao processo de detecção de agrupamentos.

3.1.2 Restrições em Nível de Atributo

Restrições em *nível de atributo* são um tipo de restrição que traz benefícios adicionais quando comparado com restrições em nível de instância. A representação da informação adicional se apresenta de maneira mais informativa quando expressada em nível de atributo, uma vez que se pode considerar mais informações do conjunto de dados. Como por exemplo, em (SCHMIDT; BRANDLE; KRAMER, 2011; PENSA et al., 2010) são apresentadas novas formas de utilização das restrições *must-link* e *cannot-link*, que até então são aplicadas somente em relação as instâncias. Essas diferentes formas serão mais detalhas a seguir.

No trabalho apresentado em (PENSA et al., 2010), o usuário ou especialista do domínio cria restrições *must-link* e *cannot-link* para instâncias e para atributos. Criar restrições que indiquem que dois atributos distintos compartilhem características semelhantes e que forcem instâncias a ficarem juntas (*must-link*) ou caso contrário, forcem a ficarem separadas (*cannot-link*), é uma particularidade usual para o domínio de expressões gênicas abordado por este trabalho. Quando o algoritmo gera os particionamentos baseando-se nas restrições, podem ser considerados diferentes grupos de atributos para determinados grupos de instâncias. Isto é, as restrições têm o papel de auxiliar a detecção de grupos tanto para instâncias quanto para atributos.

Para exemplificar essa abordagem, considere a Figura 3.10, em que os atributos (colunas) são denotados por amostras biológicas e cada instância (linhas) está associada a um particular gene. Por exemplo, o valor da expressão gênica para o gene 2 no experimento 3 é 5. Todas as instâncias do grupo delimitadas pela linha contínua compartilham valores semelhantes para os atributos destacados em negrito. Da mesma forma ocorre para o grupo de instâncias delimitadas pela linha pontilhada em relação aos atributos também destacados em negrito. Essa tarefa de agrupamento é nomeada por *co-agrupamento*, e é uma tarefa que fornece uma *bi-partição*, ou seja, duas partições fortemente relacionadas, constituída por *co-grupos*. Cada co-grupo é formado por um grupo de instâncias associadas a um grupo de atributos. Na visão do autor, esse tipo de associação pode contribuir para novas percepções de especialistas. Além disso, esse co-agrupamento tem o objetivo de aumentar a qualidade do agrupamento dos dados, uma vez que ele tenta encontrar, por meio de heurística, qual o melhor grupo de atributos para determinado grupo de instâncias.

Em outro modelo de restrições, uma adaptação das restrições em nível de instância *must-link* e *cannot-link* é proposta em (SCHMIDT; BRANDLE; KRAMER, 2011). Neste caso, a sua essência é agrupar instâncias por meio de valores dos atributos. As restrições propostas são chamadas de *must-Link* (mL) e *must-Link-Excl* (mLx), sendo que uma restrição mL descreve quais instâncias devem ser agrupadas juntas devido as características de seus atributos. Já uma restrição mLx define tanto as características das instâncias que devem ser agrupadas juntas, quanto as que não devem pertencer ao mesmo grupo.

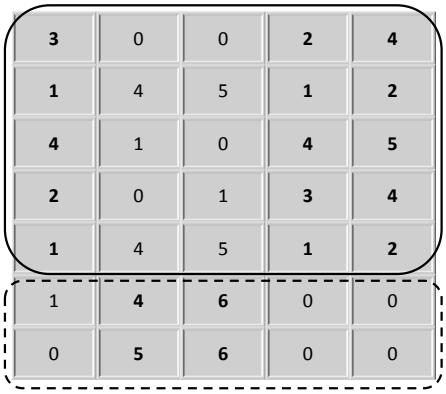

Figura 3.10 – Conjunto de expressões gênicas de amostras biológicas, adaptado de (PENSA et al., 2010).

Nesse trabalho, que considera apenas dados binários, o conjunto de restrições em nível de atributo é formado por restrições representadas por fórmulas em lógica proposicional sobre os conjuntos de atributos que descrevem cada instância.

3.1.3 Restrições em Nível de Grupo

Vários trabalhos da literatura na área mostram que restrições em nível de instância podem melhorar de forma significativa o desempenho de processos de detecção semi-supervisionada de agrupamentos comparado com processos totalmente não-supervisionados. Tais restrições têm a característica de serem independentes umas das outras e das instâncias do conjunto de dados. Desta forma, tais restrições independentes, exigidas pelo modelo em nível de instância, só podem ser fornecidas de forma eficiente quando o usuário pode visualizar todo o espaço de dados (DUBEY; BHATTACHARYA; GODBOLE, 2010). Contudo, a visualização dos dados é difícil, se não impossível em alta dimensionalidade. Além disso, algumas abordagens limitam a forma com que o uso de observações (*feedbacks*) podem auxiliar o agrupamento dos dados, ou seja, as abordagens estão limitadas a construir grupos somente sob restrições. Essas questões motivaram a proposta do uso de restrições em *nível de grupo* para que o usuário não tenha limitações em modificar restrições em meio ao processo de agrupamento (DUBEY; BHATTACHARYA; GODBOLE, 2010; MUELLER; KRAMER, 2010).

Em (DUBEY; BHATTACHARYA; GODBOLE, 2010) foi proposto um *framework* de semi-supervisão em nível de grupo, interativo, para gerar o agrupamento dos dados, onde tais restrições podem ser fornecidas pelo usuário. Algoritmos de agrupamento baseados nesse modelo normalmente iteram sobre dois passos: atribuir dados aos grupos; e ajustá-los para minimizar a distorção. Nesse modelo, o usuário fornece dois tipos diferentes de *feedback*, que terão o papel de restrições e têm como objetivo orientar o processo de supervisão sob esses dois passos enquanto o algoritmo executa. Os tipos de *feedback* são:

- Usando *assignment feedback*, o usuário move uma instância de um dos grupos atuais para outro;
- Usando *cluster description feedback*, o usuário modifica o vetor de características de qualquer grupo atual para fazê-lo mais significativo.

Com isso, a cada iteração o algoritmo aprende com o *feedback* do usuário, levando em conta outros *feedbacks* fornecidos em estágios anteriores e reagrupa o conjunto de dados até que atinja a satisfação do usuário. Por exemplo, considere uma grande base de dados sobre postagens em um lista de discussão relacionada a veículos, e que um analista deseja entender sobre o que está sendo discutido. Para isso, ele resolve particionar as postagens em k grupos, usando um algoritmo completamente não-supervisionado. O resultado dessa partição pode ser visto na Figura 3.11(a). As postagens foram divididas em dois grupos, sendo que $C1$ é sobre $\{Yamaha, Honda, car, bike, GM\}$ e $C2$ sobre $\{parts, power-steering, door, power-windows\}$. Usando seu conhecimento de domínio, o analista rotula os grupos: $C1$ como *Bikes & Cars* e $C2$ como *Car Parts*.

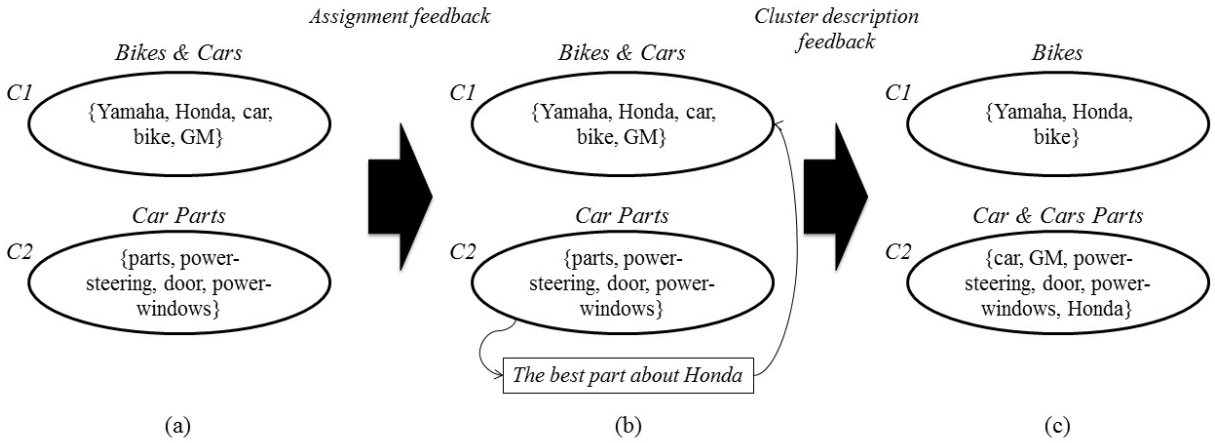

Figura 3.11 – Exemplo de um agrupamento de postagens sobre veículos, adaptado de (DUBEY; BHATTACHARYA; GODBOLE, 2010).

Na primeira iteração, o analista percebe que a postagem "*the best part about Honda*" foi atribuída incorretamente ao grupo $C2$. Então ele corrige esse erro mudando a instância de um grupo para outro, o que é chamado de *assignment feedback*, e a partir disso o algoritmo de agrupamento aprende com o *feedback* para que uma postagem similar seja atribuída corretamente ao grupo posteriormente, como mostra a Figura 3.11(b). Já em uma segunda iteração, o analista deseja que o grupo $C1$ seja sobre *Bikes* e o grupo $C2$ sobre *Cars & Car Parts*. Para isso, ele ajusta a descrição dos grupos, mudando a descrição de $C1$ para $\{Yamaha, Honda, bike\}$ e de $C2$ para $\{car, GM, power-steering, door, power-windows, Honda\}$, mostrado na Figura 3.11(c).

Essa alteração é chamada de *cluster description feedback*, onde o usuário modifica as características de descrição dos grupos de acordo com a sua preferência, e com isso o algoritmo aprende com o *feedback* e reatribui as postagens para os grupos apropriados.

Para formalizar essa abordagem, tem-se dois conjuntos de *feedbacks*, F_l^{af} de l *assignment feedbacks* e F_k^{df} de k *cluster description feedbacks*, fornecido por usuários em diferentes estágios do processo interativo. Isso ocorre da seguinte forma, o i-ésimo *assignment feedback* pode ser representado como $f_i^{af}\left(x_i^{af}, C_i^{af}, c_i^{af}\right)$, indicando que a instância x_i^{af} é atribuída pelo usuário para um grupo específico atual c_i^{af} de um conjunto de grupos anteriores C_i^{af}. Semelhantemente, para o i-ésimo *cluster description feedback*, assume-se que o usuário observa as m características mais relevantes de um grupo, ordenadas por peso o_i^{df}, e fornece seu vetor de características preferido p_i^{df} como *feedback*. De forma que, $f_i^{df}\left(p_i^{df}, o_i^{df}\right)$, especifica que p_i^{df} é um conjunto ordenado de características e o_i^{df} é o vetor de pesos sobre as características.

Outro exemplo de restrição em nível de grupo é apresentado em (MUELLER; KRAMER, 2010), onde é abordado o problema de encontrar um conjunto de grupos $\Pi = \{C_1, ..., C_k\}$ em um conjunto de dados de imagens. Para isso, são usados dados estruturados (metadados) dessas imagens e restrições $R(C)$ definidas pelo usuário, gerando assim, um conjunto de grupos *candidatos* $\Pi' = \{C'_1, ..., C'_l\}$, sendo $k < l$, no qual esses grupos candidatos são potencialmente relevantes para o agrupamento, de forma que $\Pi \subseteq \Pi'$. O processo se resume em combinar grupos candidatos a fim de otimizar uma função objetivo refletindo a qualidade desse agrupamento.

Em (MUELLER; KRAMER, 2010), existem basicamente três tipos de restrições, que depois de informadas, são mapeadas para modelos de programação linear inteiros para serem aplicadas ao processo de detecção de agrupamentos. Dentre esses tipos estão as restrições *set-level*, *clustering* e *optimization*, cada uma delas com a seguinte função: uma restrição *set-level* está na forma de fórmulas lógicas e controlam quais grupos (candidatos) devem ou não estar juntos; uma restrição *clustering* define características básicas do agrupamento, como grau de completude, grau de sobreposição, determina se um grupo pode ser combinado com outros, restringe o número de grupos total e pode limitar o número de vezes que uma instância pode ser coberta por grupos; e uma restrição *optimization* determina a função objetivo em relação a qualidade do agrupamento para ser otimizada.

Ambas as abordagens apresentadas nesta seção enfatizam a interação do usuário com o processo de detecção e aperfeiçoamento de agrupamentos de um conjunto de dados. Vale ressaltar que para esse tipo de técnica uma boa e intuitiva interface com o usuário é imprescindível, já que em alguns casos pode-se trabalhar com grandes conjuntos de dados.

3.1.4 Restrições Relativas

A restrição relativa é dada considerando que entre três instâncias, duas delas formam um par mais próximo. Em (KUMAR; KUMMAMURU, 2008) esse tipo de restrição diz

respeito as comparações relativas à distância entre três instâncias, por exemplo, dadas três instâncias x_i, x_j e x_k, uma restrição relativa diz que "x_i *está mais próximo de* x_j *do que de* x_k". Esse tipo de representação é chamada pelo autor de *restrições triplas*, e é abordado em um contexto textual. Para formalizá-la considere um conjunto de instâncias $X = \{x_1, ..., x_n\}$ de dados não-rotulados. A comparação relativa é assumida em termos de triplas $\tau_j = (x_1, x_2, x_3)$, sendo que $j = 1, 2, ..., l$, indicando que x_1 é mais similar a x_2 do que x_3.

Uma outra forma de representar restrições relativas, que é consideravelmente compacta e simples em relação as restrições triplas, foi apresentada em (LIU; ZHANG; WANG, 2011) no domínio de dados numéricos. Nesse trabalho, dadas três instâncias x_i, x_j e x_k, uma representação na forma de $x_i x_j | x_k$ diz que o par de instâncias (x_i, x_j) está mais próximo em relação a instância x_k. Essa restrição é um tipo de informação adicional que indica ao processo de agrupamento que as instâncias x_i e x_j têm maior prioridade para serem atribuídas a um mesmo grupo em relação a instância x_k. Ou seja, $x_i x_j | x_k$ deve representar que $\delta(x_i, x_j) < \delta(x_i, x_k)$ e $\delta(x_i, x_j) < \delta(x_j, x_k)$, sendo $\delta()$ uma função de distância. Um exemplo desse tipo de restrição é mostrado na Figura 3.12(a), a qual é denominada de *rooted triplet*, em que cada nó folha representa uma instância e cada nó interno representa um grupo de instâncias que são mais similares em um certo nível. A Figura 3.12(b) apresenta um outro exemplo de quatro instâncias x_i, x_j, x_k e x_l, em uma hierarquia representada por quatro restrições $x_i x_j | x_k$, $x_i x_j | x_l$, $x_k x_l | x_i$ e $x_k x_l | x_j$. Assim, cada restrição relativa é igual a duas comparações relativas entre pares de instância de um mesmo conjunto de três instâncias.

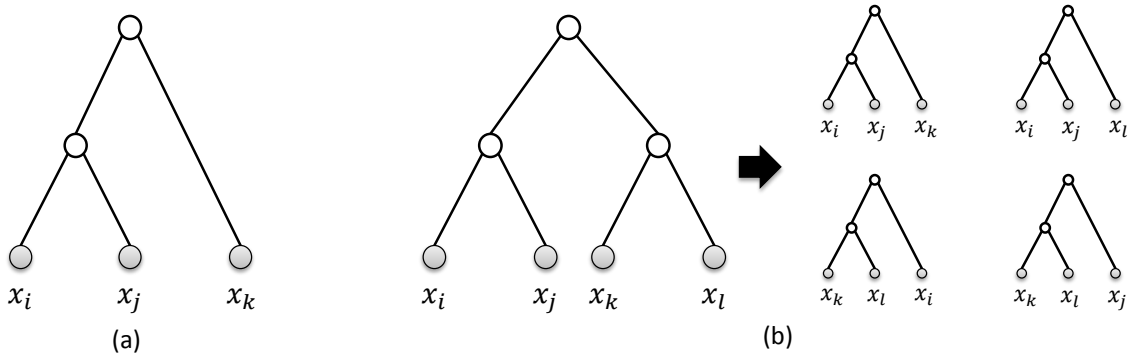

Figura 3.12 – Restrições relativas representadas graficamente. (a) Representa uma hierarquia local da restrição $x_i x_j | x_k$. (b) Um exemplo de hierarquia para quatro instâncias representadas por diversas restrições, adaptado de (LIU; ZHANG; WANG, 2011).

Uma particularidade desse tipo de restrição é a não ambiguidade em representar o conhecimento em grupos já estruturados. Por exemplo, seja uma restrição relativa $x_i x_j | x_k$, se (x_i, x_k) ou (x_j, x_k) já pertencerem a um mesmo grupo, então x_i, x_j e x_k devem todos

pertencer ao mesmo grupo. Perceba que as três instâncias podem ser escolhidas arbitrariamente de um conjunto de todas as instâncias, isso implica que as instâncias de uma determinada restrição podem pertencer a um mesmo grupo ou a três grupos distintos.

Comparadas com as restrições em nível de instância, as restrições relativas trazem mais informatividade em relação ao conhecimento representado por ambas restrições. Mesmo que seja fácil gerar restrições entre pares de instâncias (*must-link* e *cannot-link*) de um conjunto de amostras rotuladas, o número de restrições que pode ser obtido é consideravelmente menor que o número de restrições relativas. Ou seja, a partir de um conjunto de amostras rotuladas como um conjunto de treinamento, as restrições relativas capturam muito mais informações do que restrições entre instâncias. Por exemplo, uma restrição *must-link* representa duas instâncias que devem pertencer ao mesmo grupo, entretanto, se as restrições foram geradas a partir de rótulos de classes, essa restrição pode não representar uma informação correta se for gerado mais de um grupo para uma determinada classe. De forma semelhante, esse problema se estende para a restrição *cannot-link*, onde pode ocorrer de duas instâncias estarem em grupos incorretos e mesmo assim a restrição poder ser satisfeita (KUMAR; KUMMAMURU, 2008).

A principal vantagem no uso de restrições relativas em relação aos vários modelos de restrições existentes é que elas integram o conhecimento de restrições em nível de instância com conhecimento em nível de estrutura. Usualmente, restrições em nível de instância se encaixam bem em esquemas de agrupamento por particionamento, como por exemplo o *k-means*, enquanto que o seu uso em agrupamento hierárquico não é muito intuitivo. Entretanto, cada restrição relativa, que funciona como uma extensão das comparações relativas em nível de instância, representa uma hierarquia local, de forma que um par de instâncias mais próximo é agrupado antes do que as instâncias de maior distância (Figura 3.12(a)). Dessa forma, como mostra a Figura 3.13, as restrições relativas fornecem uma representação do conhecimento para *frameworks* de agrupamento por particionamento a partir de uma estrutura hierárquica(LIU; ZHANG; WANG, 2011).

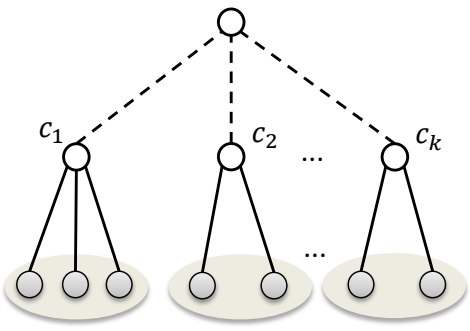

Figura 3.13 – Representação de um agrupamento por particionamento dentro de uma hierarquia, adaptado de (LIU; ZHANG; WANG, 2011).

3.1.5 Restrições por *Ranking*

Abordagens semi-supervisionadas têm se mostrado eficientes em tarefas de agrupamento, de modo que o usuário possa usar seu conhecimento de domínio, em forma de restrições, para aumentar a qualidade dessa tarefa. Entretanto, a participação do usuário nesse processo, em alguns casos, é um tanto quanto limitada. Por exemplo, as restrições *must-link* e *cannot-link* não fornecem muita liberdade para expressar o conhecimento, de forma que só é possível dizer se um par de instâncias deve ou não pertencer ao mesmo grupo.

Em certos casos, o usuário pode preferir que algumas restrições sejam aplicadas primeiro que outras, ou seja, permitir que restrições sejam atendidas por ordem de importância, pois, existe a possibilidade de que um conjunto de dados tenha múltiplos agrupamentos possíveis. Para abordar esse problema, em (AHMED; NABLI; GARGOURI, 2012) é proposto um novo tipo de restrição que ranqueia restrições do tipo *must-link* e *cannot-link* entre grupos pelo nível de sua importância.

As restrições por *ranking* podem ser classificadas em dois tipos. A primeira, trata de uma abordagem qualitativa, onde o ranqueamento da restrição deve ser relativa a preferência do usuário, por exemplo, o usuário prefere a restrição r_i à restrição r_j. A segunda trata de uma abordagem quantitativa, onde o ranqueamento das restrições é especificado usando uma função de pontuação que associa uma pontuação numérica a todas as restrições. Dentre esses dois tipos, a abordagem qualitativa é implementada em (AHMED; NABLI; GARGOURI, 2012; AHMED; NABLI; GARGOURI, 2013), e foi incorporada em um processo hierárquico de agrupamento semi-supervisionado em bases de dados textuais.

O modelo de restrições por *ranking* é definido de forma que, dados dois grupos C_i e C_j pertencentes ao conjunto de grupos Π, uma restrição é um relacionamento entre C_i e C_j, denotado por $r\left(C_i, C_j\right)$. Dado um conjunto de restrições R, um conjunto de restrições *ranking* R_k é uma ordem parcial estrita $R_k = (R, >)$, onde $(r_i > r_j)$ diz que r_i tem mais importância que r_j. Pode-se perceber que as restrições se assemelham a um princípio comum, onde pessoas frequentemente expressam a sua classificação (*ranking*) em termos como: "*eu considero que A é mais importante que B*". Esse tipo de ranqueamento é bastante intuitivo e pode ser amplamente aplicado. Nesse trabalho o processo começa criando grupos individuais para cada instância, e se resume a unir esses grupos por ordem de ranqueamento das restrições (*must-link* e *cannot-link*).

3.2 Incorporando Restrições ao Agrupamento

Dado um conjunto de restrições é preciso incorporá-lo ao processo de detecção de agrupamentos. Essa seção destaca as duas principais estratégias para que o modelo de

restrição possa ser utilizado com êxito, que são: baseada em busca (Seção 3.2.1) e baseada em medidas de similaridade (Seção 3.2.2).

3.2.1 Baseada em Busca

Este tipo de estratégia modifica a busca a cada iteração a fim de encontrar agrupamentos apropriados para o conjunto de dados sobre um conjunto de instâncias que interferem na minimização de uma função objetivo. A ideia geral é orientar a atribuição das instâncias aos grupos mais adequados a cada iteração do algoritmo, verificando a todo momento um conjunto de restrições informado.

Algoritmo 2 - `COP-kmeans`

Entrada: Conjunto de dados X, Restrições *must-link* R_{ml}, Restrições *cannot-link* R_{cl}, Número de grupos k
Saída: Partições $\Pi = \{C_1, ..., C_k\}$

```
Seleciona aleatoriamente μ_1, ..., μ_k como centróides iniciais
repita
  para cada instância x_i ∈ X faça
    Atribui x_i ao grupo C_j mais próximo tal que
      nenhuma restrição em R_ml e R_cl seja violada
    se não conseguir atribuir a nenhum grupo C_j então
      O algoritmo falha!
      retorna Π = ∅
    fim se
  fim para
  para cada grupo C_i faça
    Atualiza o centróide μ_i pela média de todas as instâncias x_j atribuídas a ele
  fim para
até convergir
retorna Π = {C_1, ..., C_k}
```

Para ilustrar esse processo, considere o algoritmo COP-*kmeans* descrito em (WAGSTAFF et al., 2001). Esse algoritmo se trata de uma adaptação do *k-means* para detecção semi-supervisionada de agrupamentos (veja Algoritmo 2). Para tanto, o COP-*kmeans* recebe como entrada um conjunto de dados X, um número de grupos desejado k e conjuntos de restrições *must-link* (R_{ml}) e restrições *cannot-link* (R_{cl}). O primeiro passo do algoritmo é inicializar os k centróides, da mesma forma como é feito no tradicional *k-means* (apresentado na Seção 2.2). A partir disso, cada instância é atribuída ao centróide mais próximo, desde que não seja violada nenhuma restrição. Após esta etapa, são atualizados novamente os k centróides pela média das instâncias de cada grupo e repete-se os passos anteriores até convergir, ou seja, até que os centróides se estabilizem.

Um detalhe importante a ser destacado é que, se em alguma iteração o COP-*kmeans* não conseguir atribuir uma instância a nenhum grupo, devido a conflitos de restrições, então o algoritmo terá a sua execução interrompida e não será retornada nenhuma partição

dos dados. A Figura 3.14 ilustra os casos que levam o algoritmo COP-*kmeans* a falha. Primeiramente, considerando um conjunto de dados particionado em dois grupos, no qual as instâncias x_i e x_j foram atribuídas a grupos distintos no decorrer do processo e a instância x_k está para ser atribuída a algum grupo. No caso da Figura 3.14(a), nota-se duas restrições *cannot-link*, $r_{cl}(x_i, x_k)$ e $r_{cl}(x_j, x_k)$, que provocarão a falha do algoritmo, pois a instância x_k não poderá ser atribuída a nenhum grupo. De forma semelhante, a Figura 3.14(b) apresenta as restrições *must-link*, $r_{ml}(x_i, x_k)$ e $r_{ml}(x_j, x_k)$, o que também provocará a falha, pois a instância x_k deveria pertencer aos dois grupos simultaneamente. Outro caso, apresentado na Figura 3.14(c), ocorre quando duas instâncias (x_i e x_j) estão em um mesmo grupo e suas restrições, em relação a uma terceira instância (x_k), atuam de forma contraditória. Nesse caso, a instância x_k não será atribuída a nenhum dos grupos existentes. Durante o processo de agrupamento isso pode ocorrer algumas vezes, e com maior frequência quando há um grande número de restrições.

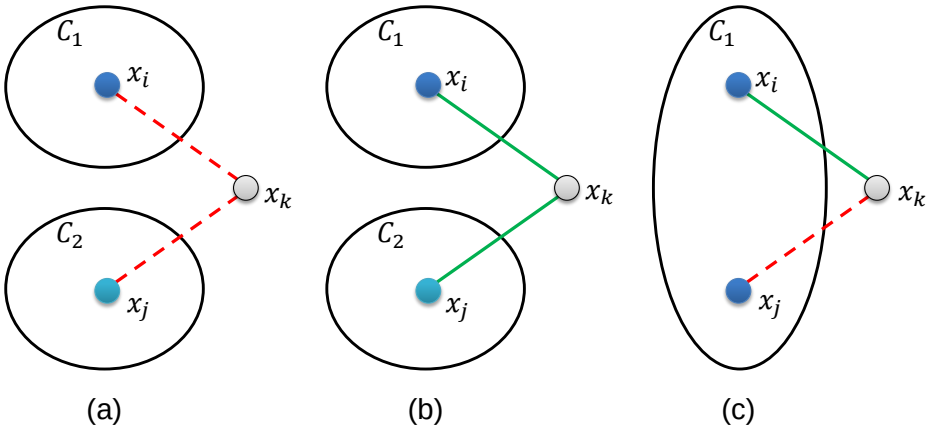

Figura 3.14 – Exemplos de casos onde o algoritmo COP-*kmeans* pode falhar, adaptado de (HUANG; CHENG; ZHAO, 2008).

Para amenizar esse problema, o trabalho descrito em (HUANG; CHENG; ZHAO, 2008) propõe um algoritmo que aproveita de melhor forma a informação fornecida pelas restrições, de maneira que, um centróide auxiliar é gerado para cada grupo dentro do processo de agrupamento. Esse algoritmo é denominado MLC-*kmeans* (veja Algoritmo 3), e também se trata de uma adaptação do algoritmo *k-means*. O MLC-*kmeans* consegue particionar as instâncias que participam de restrições *must-link* e usar a informação espacial dessas instâncias para gerar um novo representante auxiliar para cada um dos k grupos. De forma que, durante o processo de agrupamento, cada grupo irá contar com dois representantes, um baseado nas restrições e o outro baseado em centróides derivados do algoritmo *k-means*. A partir disso, a cada atribuição de instância a algum grupo, será calculada a distância para os dois representantes, atribuindo a instância ao grupo de representante mais próximo, desde que não viole nenhuma restrição *cannot-link*. Perceba que, ao verificar somente as restrições *cannot-link*, a possibilidade de conflitos entre restri-

ções diminui consideravelmente, pois, neste caso, somente situações como a apresentada na Figura 3.14(a) irão provocar falha durante o processo do algoritmo.

Algoritmo 3 - `MLC-kmeans`

Entrada: Conjunto de dados X, Restrições *must-link* R_{ml}, Restrições *cannot-link* R_{cl}, Número de grupos k
Saída: Partições $\Pi = \{C_1, ..., C_k\}$

```
Seleciona aleatoriamente μ_1, ..., μ_k como centróides iniciais
repita
  para cada par (x_i, x_j) ∈ R_ml faça
    Atribui x_i e x_j para o grupo mais próximo
  fim para
  para cada grupo C_j faça
    Atualiza o centróide a_j pela média de todas as instâncias x_i ∈ R_ml atribuídas a ele
  fim para
  para cada instância x_i ∈ X faça
    Atribui x_i ao grupo C_j com representante mais próximo tal que
      nenhuma restrição em R_cl seja violada
    se não conseguir atribuir a nenhum grupo C_j então
      O algoritmo falha!
      retorna Π = ∅
    fim se
  fim para
  para cada grupo C_i faça
    Atualiza o centróide μ_i pela média de todas as instâncias x_j atribuídas a ele
  fim para
até convergir
retorna Π = {C_1, ..., C_k}
```

Outro detalhe a respeito de algoritmos como o COP-*kmeans* e o MLC-*kmeans* é que eles adotam uma abordagem *hard* para as restrições, isto é, não permitem que as restrições sejam violadas em nenhuma hipótese. Enquanto isso, uma abordagem *soft* para restrições permite que restrições sejam violadas, acrescentando uma penalidade para cada violação, como implementado no algoritmo PC-*kmeans* (BILENKO; BASU; MOONEY, 2004). Nessa abordagem o algoritmo tenta minimizar uma função objetivo composta pela soma das distâncias entre as instâncias e seus centróides representantes, o custo de violação das restrições *must-link* e o custo de violação das restrições *cannot-link*. O cálculo da função objetivo é dada por:

$$f_{pckm} = \sum_{j=1}^{k} \sum_{x_i \in c_j} \|x_i - \mu_j\|^2 + \sum_{(x_i,x_j)\in R_{ml}} w_{ij} + \sum_{(x_i,x_j)\in R_{cl}} \bar{w}_{ij}$$

sendo que w_{ij} é o valor de custo da violação de uma restrição *must-link* $R_{ml}\,(x_i, x_j)$, e $\bar{w}_{ij}$ é o valor de custo da violação de uma restrição *cannot-link* $R_{cl}\,(x_i, x_j)$.

3.2.2 Baseada em Medidas de Similaridade

A estratégia para algoritmos de semi-supervisão que alteram a medida de similaridade tem como base uma determinada função objetivo, que ao longo do processo de agrupamento tenta ser minimizada ou maximizada. Desse modo, o espaço de dados pode ser alterado para que, por exemplo, uma determinada função objetivo aproxime instâncias que devem estar juntas em um mesmo agrupamento e afaste as instâncias que não devem estar juntas, por meio das restrições, alterando a distância das instâncias do conjunto de dados que inicialmente são baseadas em suas características.

Esse modo de usar o conhecimento das restrições para aprimorar o agrupamento final é descrito com detalhes em (BASU; DAVIDSON; WAGSTAFF, 2008). Considerando um conjunto de dados X, um conjunto de restrições *must-link* R_{ml} e um conjunto de restrições*cannot-link* R_{cl}, a tarefa do agrupamento semi-supervisionado é encontrar uma distância métrica δ que minimize a equação 3.16 e maximize a equação 3.17. Após ter o espaço métrico ajustado conforme as restrições, um algoritmo não-supervisionado poderá ser aplicado para gerar o agrupamento do conjunto de dados.

$$\sum_{r_{ml}(x_i,x_j)\in R_{ml}} \delta\left(x_i, x_j\right) \tag{3.16}$$

$$\sum_{r_{cl}(x_i,x_j)\in R_{cl}} \delta\left(x_i, x_j\right) \tag{3.17}$$

Em (LIU; JIN; JAIN, 2007), é proposto um algoritmo que trabalha sob uma função objetivo apropriada para identificar o subespaço que mantém as instâncias de pares de restrições *must-link* insatisfeitas mais próximas umas das outras e mantém as instâncias de pares de restrições *cannot-link* insatisfeitas mais afastadas. Com isso, o algoritmo tem o objetivo de identificar e minimizar o número de restrições insatisfeitas a cada iteração, contribuindo para um agrupamento de dados mais aprimorado.

A Figura 3.15 apresenta um exemplo da abordagem adotada em (LIU; JIN; JAIN, 2007). Considerando um conjunto de dados, um conjunto de restrições *must-link* (linhas sólidas) e um conjunto de restrições *cannot-link* (linhas pontilhadas) dadas como entrada para o algoritmo (Figura 3.15(a)), a cada iteração, o algoritmo modifica o espaço de dados para que, por meio das restrições, as instâncias mais similares fiquem mais próximas e as mais dissimilares fiquem mais distantes. A representação dessas alterações pode ser vista das Figuras 3.15(b) à 3.15(d). Outros algoritmos que utilizam esta abordagem podem ser encontrados em (CHAPELLE; SCHÖLKOPF; ZIEN, 2006).

Até aqui, foram apresentados diversos tipos de restrições empregados em diferentes domínios e também algumas formas simples de aplicação dessas restrições ao conjunto de dados. A Tabela 1.4 resumi os tipos de restrições para a semi-supervisão aos trabalhos citados ao longo deste capítulo. Contudo, as restrições também podem ter um propósito fundamental ao realizar a avaliação dos particionamentos resultantes dos algoritmos. Para

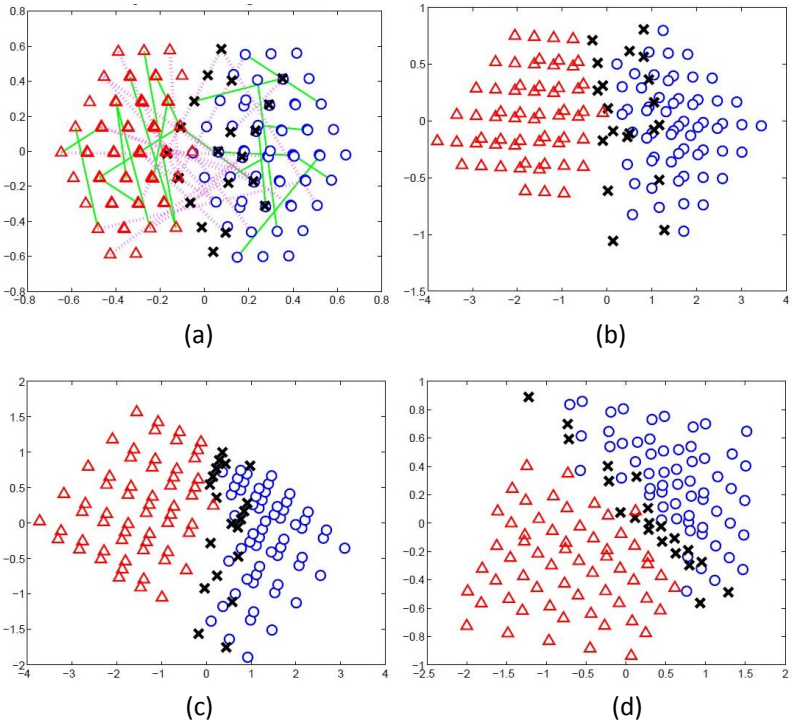

Figura 3.15 – Exemplo de alteração do espaço de pontos de dados. (a) Distribuição do conjunto de dados original. (b) - (d) Transformações do espaço de dados em iterações do algoritmo. Adaptado de (LIU; JIN; JAIN, 2007).

tal, existe uma metodologia de avaliação específica para algoritmos de semi-supervisão, a qual é apresentada logo a seguir.

3.3 Metodologia de Avaliação para Semi-Supervisão

O trabalho descrito em (POURRAJABI et al., 2014) aborda uma nova metodologia de avaliação para algoritmos de detecção de agrupamentos de dados que tem como característica o uso de informações de semi-supervisão. O principal problema abordado neste trabalho diz respeito a forma com que a evolução de métodos de avaliação não acompanha a evolução dos algoritmos de agrupamento. Tendo em vista que, algoritmos de semi-supervisão têm sido amplamente estudados nos últimos tempos e, no entanto, não havia uma metodologia especifica para compará-los de forma eficiente.

A ideia principal desta nova metodologia é apontar qual dos algoritmos ou qual das configurações de um determinado algoritmo melhor se ajusta a informação adicional, dada por restrições, tendo como perspectiva a estimativa do **erro de classificação**[1] baseado em um procedimento de *cross-validation*. Isto é, uma vez que um algoritmo de agrupamento fornece uma rotulagem relativa em vez de rotulagem absoluta dos dados,

[1] O erro de classificação é a fração de predições incorretas para um classificador considerando um conjunto de teste.

Tabela 3.1 – Resumo dos diferentes tipos de restrições apresentados neste capítulo.

Tipo de Restrição	**Abordagem de Agrupamento**	**Domínio de Dados**	**Referências**
Nível de Instância	Baseada em Busca e Medida de Similaridade	Texto; Imagem; Numérico; Nominal.	(WAGSTAFF; CARDIE, 2000) (JIANG et al., 2013) (LAI et al., 2014)
Nível de Atributo	Baseada em Busca	Binário; Numérico.	(SCHMIDT; BRANDLE; KRAMER, 2011) (PENSA et al., 2010)
Nível de Grupo	Baseada em Busca	Texto; Imagem.	(DUBEY; BHATTACHARYA; GODBOLE, 2010) (MUELLER; KRAMER, 2010)
Relativa	Baseada em Medidas de Similaridade	Texto; Numérico.	(KUMAR; KUMMAMURU, 2008) (LIU; ZHANG; WANG, 2011)
Ranqueada	Baseada em Busca	Texto.	(AHMED; NABLI; GARGOURI, 2012) (AHMED; NABLI; GARGOURI, 2013)

a nova metodologia ajusta as informações de semi-supervisão disponíveis para estimar corretamente um erro de classificação.

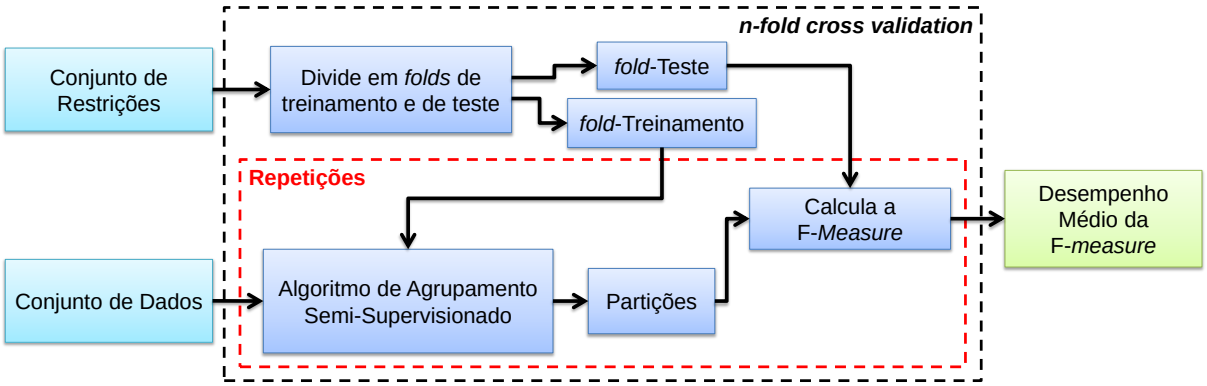

Figura 3.16 – Ilustração do processo de avaliação de algoritmos de semi-supervisão baseado na metodologia apresentada em (POURRAJABI et al., 2014).

A Figura 3.16 ilustra as etapas dessa metodologia de avaliação. Em um processo *n-fold cross validation* divide-se o conjunto de restrições em *folds* de treinamento e teste. Para n divisões do conjunto de restrições, de maneira geral, a informação disponível para o agrupamento é dada por $\left(\frac{n-1}{n}\right)$, e o restante $\left(\frac{1}{n}\right)$ será a informação disponível para avaliação. Em caso de algoritmos de inicialização aleatória, como por exemplo, algoritmos derivados do *k-means*, podem haver várias repetições de execuções de um mesmo *fold* durante o processo, amenizando assim a natureza não-determinística desses algoritmos.

Entretanto, no contexto de agrupamento semi-supervisionado alguns detalhes devem ser considerados para que seja realizada uma avaliação apropriada. Primeiramente, deve-se garantir que a mesma informação de semi-supervisão não seja utilizada em ambos os processos de agrupamento e avaliação, o que pode subestimar o determinado erro de classificação. Este problema será melhor abordado na Seção 3.3.1.

Outra questão é como elaborar uma medida para calcular o erro de classificação. Neste caso, para medir e comparar o desempenho dos algoritmos quantitativamente, deve-se transformar o problema de agrupamento semi-supervisionado em um problema de classificação sobre as restrições. Para isso, pode-se usar a medida F-*measure* amplamente conhecida na literatura de avaliação de classificadores. Na Seção 3.3.2 serão apresentados mais detalhes.

3.3.1 Independência entre os Conjuntos de Treinamento e Teste

O problema associado ao *cross-validation*, ou qualquer outro procedimento de avaliação que divide a informação disponível em partições de treinamento e teste, pode ser visto mais facilmente considerando um conceito de **transitividade** entre as restrições. Considere um conjunto de instâncias e um conjunto de restrições representados por um grafo, no qual as instâncias são os vértices e as restrições são as arestas. Por meio da transitividade, novas arestas (restrições) podem ser induzidas a partir das arestas previamente informadas. Por exemplo, considerando as instâncias x_i, x_j, x_k, x_l e as restrições $r_{ml}(x_i, x_j)$, $r_{ml}(x_k, x_l)$ e $r_{cl}(x_j, x_k)$, como ilustrado na Figura 3.17(a), pode-se então, induzir as restrições $r_{cl}(x_i, x_k)$, $r_{cl}(x_i, x_l)$ e $r_{cl}(x_j, x_l)$ (Figura 3.17(b)). Embora a transitividade gere uma quantidade considerável de arestas, isso não implica que sempre será induzida uma nova aresta para todo par de vértices. Ou seja, ao inverter as restrições do exemplo anterior para $r_{cl}(x_i, x_j)$, $r_{cl}(x_k, x_l)$ e $r_{ml}(x_j, x_k)$, as restrições $r_{cl}(x_i, x_k)$ e $r_{cl}(x_j, x_l)$ podem ser derivadas, mas nada pode ser inferido entre o par (x_i, x_l).

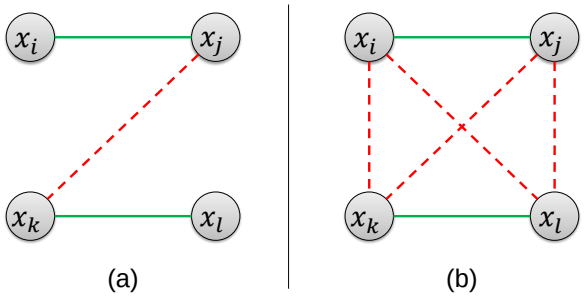

Figura 3.17 – Ilustração da indução de novas restrições por meio da transitividade, adaptado de (POURRAJABI et al., 2014).

Dessa forma, a transitividade entre restrições pode conduzir, não intencionalmente, a presença indireta de informação em algum *fold*. Para exemplificar, suponha um *fold* de

treinamento que contêm as restrições $r_{ml}(x_i, x_j)$ e $r_{cl}(x_j, x_k)$. Se o *fold* de teste possuir a restrição $r_{cl}(x_i, x_k)$, então tem-se o problema de compartilhamento da mesma informação nas fases de agrupamento e avaliação, pois, esta informação já estava implicitamente disponível durante o processo de agrupamento, mesmo não sendo explicitamente informada. Para abordar esse problema, o trabalho em (POURRAJABI et al., 2014) propõe a divisão do grafo de restrições, podando algumas arestas até que os *folds* não se sobreponham. Essa abordagem baseada em grafos pode fornecer uma solução para evitar esse problema em um nível abstrato.

Para garantir que o procedimento *cross-validation* não gere o problema de usar a mesma informação nos *folds* de treinamento e teste, particiona-se todas as instâncias envolvidas em qualquer restrição em seus respectivos *folds* e, então, exclui as restrições que envolvam simultaneamente instâncias do *fold* de treinamento com instâncias do *fold* de teste. Para uma abordagem *n-fold cross validation*, particiona-se as instâncias em n *folds* e, a cada iteração, utiliza-se $n-1$ *folds* como o conjunto de treinamento e as demais restrições como o conjunto de teste.

3.3.2 Avaliação do Agrupamento

Nesta etapa, utiliza-se as restrições para estimar a qualidade de uma partição produzida por um algoritmo de agrupamento. De forma que, uma partição é avaliada como um classificador que distingui a classe das restrições *must-link* da classe das restrições *cannot-link*. Isto é, avalia-se para cada par de instâncias no *fold* de teste se suas respectivas restrições foram reconhecidas pelo processo de agrupamento. Um dado particionamento gerado por um algoritmo de agrupamento fornece meios para avaliar o grau de satisfação ou violação das restrições no *fold* de teste. Para esse propósito, a métrica F-*Measure* pode ser utilizada para estimar a satisfação de um determinado particionamento.

O problema de agrupamento semi-supervisionado pode então ser considerado como um problema de classificação, como segue. Cada *fold* de teste possui um conjunto de restrições *must-link* (classe 1) e um conjunto de restrições *cannot-link* (classe 0). Uma partição gerada por um algoritmo de agrupamento satisfaz um certo número dessas restrições, da seguinte forma:

- ❏ Pares de instâncias envolvidas em uma restrição *must-link*:
  - atribuídas corretamente ao mesmo grupo:
    - *true positive* para a classe 1;
    - *true negative* para a classe 0.
  - atribuídas incorretamente a grupos distintos:
    - *false negative* para a classe 1;
    - *false positive* para a classe 0.

- ❏ Pares de instâncias envolvidas em uma restrição *cannot-link*:
  - atribuídas corretamente a grupos distintos:
    - *true positive* para a classe 0;
    - *true negative* para a classe 1.
  - atribuídas incorretamente ao mesmo grupo:
    - *false negative* para a classe 0;
    - *false positive* para a classe 1.

Com essa informação pode-se então calcular a **precisão** e **revocação** da satisfação do *fold* de teste sobre o particionamento gerado a partir de informações do *fold* de treinamento. Dessa forma, as categorias descritas acima compõem as taxas de precisão e revocação conforme apresentado nas Equações 3.18 e 3.19, respectivamente:

$$\text{Precisão} = \frac{true\ positive}{true\ positive + false\ positive} \tag{3.18}$$

$$\text{Revocação} = \frac{true\ positive}{true\ positive + false\ negative} \tag{3.19}$$

A medida F-*Measure* emprega uma média harmônica entre precisão e revocação, sendo que seu valor pode variar no intervalo [0,1], em que valores mais próximos de 1 significam maior acerto das restrições pelo particionamento em questão. O cálculo dessa medida é dado pela Equação 3.20, como segue:

$$\text{F-}Measure = 2\left(\frac{\text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}}\right) \tag{3.20}$$

3.4 Considerações Finais

Este capítulo apresentou detalhes referentes ao processo de detecção de agrupamento semi-supervisionado de dados, considerando as restrições como principal forma de semi-supervisão. No entanto, podem existir diversos tipos de restrições abordando variados tipos de problemas. Mesmo havendo vários tipos de restrições, a restrição entre pares de instâncias se tornou principal referência no contexto de semi-supervisão, além disso, vários outros tipos de restrições são definidos a partir dele. Também, foram abordados as duas principais vertentes para incorporar restrições ao processo de agrupamento. Por fim, foi apresentada uma metodologia específica para algoritmos de semi-supervisão, permitindo assim compará-los por meio de suas características principais, que são a forma como utilizam as informações de semi-supervisão. O próximo capítulo apresenta o algoritmo do trabalho desenvolvido descrito nesta dissertação, o qual utiliza informações entre pares de instâncias para formular um novo tipo de semi-supervisão.

Capítulo 4

MRS-*kmeans*

Este capítulo descreve o método de agrupamento semi-supervisionado de dados desenvolvido no trabalho descrito aqui, denominado MRS-*kmeans* (***M**ulti-**R**epresentative **S**emi-supervised k-means*), que utiliza restrições entre pares de instâncias a fim de definir centróides auxiliares e derivar restrições do tipo *must-link* e *cannot-link* sobre esses representantes auxiliares. Diferentemente dos algoritmos COP-*kmeans* e MLC-*kmeans*, que usam restrições em nível de instância para guiar a atribuição das instâncias aos grupos, o algoritmo MRS-*kmeans* usa essas restrições para definir múltiplos representantes auxiliares para cada centróide do *k-means*.

O capítulo está organizado da seguinte forma. A Seção 4.1 apresenta o processo de definição dos representantes auxiliares candidatos. A Seção 4.2 mostra como é feita a definição dos novos tipos de restrições. A Seção 4.3 detalha a etapa de refinamento usada para otimizar o uso dos representantes auxiliares. A Seção 4.4 apresenta o processo principal do método de agrupamento proposto. A Seção 4.5 descreve o custo computacional necessário para a execução do algoritmo. Por fim, a Seção 4.6 apresenta as considerações finais sobre o capítulo.

4.1 Definição dos Representantes Auxiliares

Algoritmos de agrupamentos de dados por particionamento convencionais, baseados em protótipo, usualmente utilizam um único representante para cada partição (veja Capítulo 2). No algoritmo *k-means*, por exemplo, cada grupo pode ser representado por um centróide, computado como a média em cada dimensão das instâncias que pertencem a esse grupo. Por se tratar de uma adaptação do algoritmo *k-means*, o MRS-*kmeans* continua utilizando a estratégia de centróides para representar as instâncias em cada grupo, porém, os representantes auxiliares também são centróides, mesmo sendo gerados por um tipo diferente de informação, as restrições. Portanto, para diferenciar cada um dos dois tipos de centróides existentes será adotada a seguinte terminologia:

- **centróide principal** é aquele que representa todas as instâncias em um determinado grupo, simbolizado por μ;

- **centróide auxiliar** é aquele que representa instâncias interconectadas por uma ou mais restrições *must-link*, simbolizado por a.

Antes que o funcionamento do processo de definição dos centróides auxiliares possa ser explicado, o termo **componente conexa** precisa ser definido. Em teoria de grafos, dado um grafo $G(V, A)$ dois vértices v_1 e $v_2 \in V$ são ditos serem conectados em G se existir uma cadeia em G com extremidade inicial v_1 e extremidade final v_2. A componente conexa de um vértice $v_1 \in V$ é definida pelo subgrafo gerado pelo conjunto de todos os outros vértices $v_2 \in V$ que lhe estão conectados, ou seja, a partir de um vértice v_1 é possível atingir qualquer vértice v_2 da componente conexa por meio de uma ou mais arestas (NETTO, 2006).

De maneira análoga, um conjunto de instâncias $\chi_i \subseteq X$ é dito ser um componente conexo se todas essas instâncias estiverem unidas por uma ou mais restrições *must-link*, mesmo que não estejam ligadas diretamente. Para exemplificar, considere as seguintes restrições *must-link*: $r_{ml}(x_i, x_j)$ e $r_{ml}(x_j, x_k)$. As duas restrições dizem exatamente que as instâncias x_i e x_j devem pertencer a um mesmo grupo, e que as instâncias x_j e x_k também devem pertencer a um mesmo grupo. Mesmo não estando explicito, pode-se concluir que as instâncias x_i e x_k também devem pertencer a um mesmo grupo, logo, todas as instâncias (x_i, x_j e x_k) fazem parte de uma componente conexa, ou seja, duas instâncias podem ser forçadas a estar no mesmo grupo sem que haja uma restrição que especifique isso.

O processo de definição dos centróides auxiliares é ilustrado na Figura 4.18. Considere um conjunto de instâncias X em um espaço bidimensional e restrições entre pares de instâncias, em que as instâncias estão representadas pelos pontos azuis, as restrições *must-link* por linhas verdes contínuas e as restrições *cannot-link* por linhas tracejadas vermelhas (veja a Figura 4.18(a)). O primeiro passo é detectar as componentes conexas existentes no conjunto, e a partir disso, definir um centróide para cada componente conexa, como é mostrado na Figura 4.18(b). Cada centróide auxiliar será a média em cada dimensão das instâncias pertencentes a uma respectiva componente conexa. O conjunto de centróides gerados com esse processo é chamado de conjunto de centróides auxiliares candidatos, representado pelo símbolo M_a, pois ainda serão eleitos auxiliares de fato em uma fase de refinamento que será apresentada mais adiante na Seção 4.3.

Uma propriedade importante desses representantes auxiliares está relacionada a quantidade de instâncias em uma componente conexa. Pois, componentes conexas com maior número de instâncias representam maior nível de informação semântica ao agrupamento e devem ser consideradas mais importantes do que componentes conexas que tenham menor

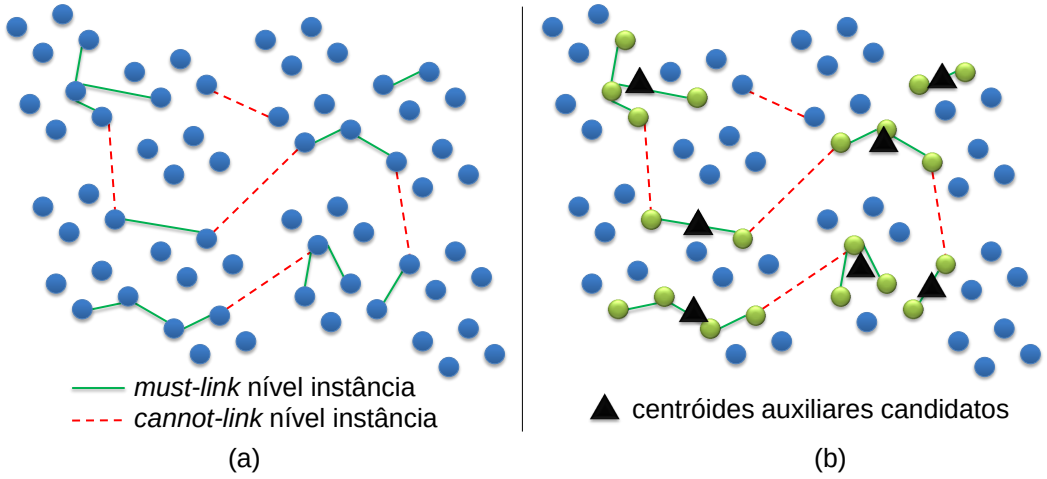

Figura 4.18 – Exemplo do processo de definição dos centróides auxiliares candidatos. (a) Conjunto de instâncias com restrições *must-link* e *cannot-link*. (b) Centróides auxiliares candidatos definidos para cada componente conexa.

número de instâncias. A Seção 4.4.1 mostra como cada centróide auxiliar será considerado durante o processo de agrupamento com relação ao seu nível de informação semântica.

Posteriormente, durante o processo de agrupamento, os centróides auxiliares candidatos devem ser atribuídos aos centróides principais com o objetivo de apoiar a definição adequada do particionamento de grupos existentes. Desta forma, auxiliando o centróide principal de cada grupo na detecção das diferentes formas existentes na verdadeira estrutura do conjunto de dados.

Com o intuito de esclarecer o funcionamento desse processo, o Algoritmo 4 apresenta os detalhes inerentes a definição de múltiplos centróides auxiliares candidatos. Considere I_m como o conjunto de todas as instâncias de X que participam de alguma restrição *must-link* $r_{ml} \in R_{ml}$, isto é, $I_m = \{x_i \mid \exists r_{ml}\,(x_i, x_j) \in R_{ml}$ tal que $x_i, x_j \in X\}$, sendo que o par de instâncias em uma restrição *must-link* r_{ml} possui a propriedade de ser independente de ordem, ou seja, $r_{ml}\,(x_i, x_j) \in R_{ml} \Rightarrow r_{ml}\,(x_j, x_i) \in R_{ml}$. Portanto, o algoritmo recebe como parâmetros de entrada um conjunto de restrições *must-link* entre instâncias R_{ml} e um conjunto M_χ de $|I_m|$ conjuntos χ, sendo que M_χ é um conjunto de componentes conexas χ das instâncias do conjunto de dados X unidas por uma ou mais restrições *must-link*. Inicialmente faz-se com que cada conjunto $\chi_l \in M_\chi$ seja composto por uma das instâncias de I_m, ou seja, $M_\chi = \left\{\{x_1\}, \{x_2\}, ..., \left\{x_{|I_m|}\right\}\right\}$. O algoritmo retorna o conjunto de centróides auxiliares candidatos M_a relativo as restrições informadas. A estratégia utilizada para a definição desses centróides auxiliares foi inspirada na estratégia descrita no trabalho apresentado em (HUANG; CHENG; ZHAO, 2008).

O Algoritmo 4 começa criando um conjunto vazio M_a (linha 1), o qual irá armazenar os centróides auxiliares candidatos gerados a partir das instâncias pertencentes a cada componente conexa χ. A primeira estrutura de repetição (linhas 2 a 10) alocará cada instância de cada restrição $r_{ml} \in R_{ml}$ a um conjunto χ_l que reuni as instâncias em

Algoritmo 4 - `RepresentantesAuxiliaresCandidatos`

Entrada: R_{ml}, $M_\chi = \left\{\{x_1\}, \{x_2\}, ..., \left\{x_{|I_m|}\right\}\right\}$
Saída: M_a
1: $M_a \leftarrow \emptyset$
2: **para** cada restrição $r_{ml}(x_i, x_j) \in R_{ml}$ **faça**
3: **se** $\exists \chi_l \in M_\chi \wedge x_i \in \chi_l$ **então**
4: $\chi_l = \chi_l \cup \{x_j\}$ /* x_j é atribuída a $\chi_l = \{x_i, x_j\}$*/
5: **senão se** $\exists \chi_l \in M_\chi \wedge x_j \in \chi_l$ **então**
6: $\chi_l = \chi_l \cup \{x_i\}$ /* x_i é atribuída a $\chi_l = \{x_j, x_i\}$*/
7: **senão se** $(\exists \chi_l \in M_\chi \wedge x_i \in \chi_l) \wedge (\exists \chi_t \in M_\chi \wedge x_j \in \chi_t)$ **então**
8: $\chi_l = \chi_l \cup \chi_t$
9: **fim se**
10: **fim para**
11: **para** cada elemento $\chi_l \in M_\chi$ **faça**
12: $a_l \leftarrow \frac{1}{|\chi_l|} \sum_{x_i \in \chi_l} x_i$
13: $M_a \leftarrow M_a \cup a_l$
14: **fim para**

sua respectiva componente conexa. Para que um par de instâncias (x_i e x_j) seja alocado corretamente a seu respectivo conjunto χ, três condições devem ser verificadas, da seguinte forma: (linha 3) se existe algum conjunto $\chi_l \in M_\chi$ e a instância x_i pertence a ele, então (linha 4) a instância x_j será atribuída a esse conjunto χ_l; de forma semelhante, (linha 5) se existe algum conjunto $\chi_l \in M_\chi$ e a instância x_j pertence a ele, então (linha 6) a instância x_i será atribuída a esse conjunto χ_l; entretanto, (linha 7) se as instâncias x_i e x_j pertencerem aos conjuntos χ_l e χ_t respectivamente, sendo $l \neq t$, então (linha 8) será feita a união desses dois conjuntos ($\chi_l \cup \chi_t$). A última estrutura de repetição (linhas 11 a 14) processa cada conjunto $\chi_l \in M_\chi$, criando um centróide a_l pela média de cada dimensão das instâncias atribuídas a esse conjunto (linha 12). Ao final, (linha 13) cada centróide a_l, que representa um conjunto de instâncias de uma componente conexa, será adicionado ao conjunto dos representantes auxiliares candidatos M_a.

Em relação a complexidade computacional do Algoritmo 4, a primeira etapa (linhas 2 a 10) será executada para cada restrição *must-link* $r_{ml} \in R_{ml}$, resultando em complexidade $O(|R_{ml}|)$. Na segunda etapa, que gera os centróides auxiliares candidatos (linhas 11 a 14), o conjunto M_χ possui l componentes conexas χ que foram detectadas na etapa anterior, e cada uma das componentes conexas χ possui um número diferente de instâncias do conjunto I_m, de forma que todas as instâncias do conjunto I_m estarão em alguma componente conexa, dessa forma, a linha 12 será executada $|I_m|$ vezes. Logo, a complexidade da segunda etapa do algoritmo é $O(|I_m|)$. Dessa forma, pode-se então definir a complexidade total do uso desse algoritmo em $O(|R_{ml}| + |I_m|)$, tendo assim um custo linear em função da quantidade de instâncias do conjunto I_m, já que a quantidade de restrições em R_{ml} é no máximo a metade da cardinalidade do conjunto I_m. Para exemplificar, considere que existam dez restrições *must-link* no conjunto R_{ml} e que cada instância participe somente

de uma dessas restrições, então o conjunto I_m terá tamanho igual a vinte, pois, cada restrição possui um par de instâncias. Nesse caso, levando em consideração que algumas instâncias geralmente participam de mais de uma restrição, o tamanho do conjunto R_{ml} será, no pior caso, no máximo a metade do tamanho do conjunto I_m, implicando que, $|R_{ml}|$ será sempre menor que $|I_m|$.

4.2 Definição do Novo Tipo de Restrição

Durante o processo de detecção de agrupamentos, os centróides auxiliares candidatos devem ser atribuídos aos grupos com centróides principais mais próximos, auxiliando dessa forma, a definição adequada de particionamento de grupos existentes no conjunto. Essa fase de atribuição, mostrada com detalhes na Secção 4.4.2, é guiada por restrições em **nível de protótipo**, isto é, restrições entre pares de representantes auxiliares.

Essa nova categoria de restrições, derivada das informações obtidas de restrições entre instâncias, é utilizada para definir se dois centróides auxiliares devem representar um mesmo grupo, ou se devem representar grupos distintos. De maneira formal, o trabalho nesta dissertação considera dois novos tipos de restrições:

- **Restrição *must-link* em nível de protótipo:** simbolizada por $r'_{ml}(a_l, a_t)$, especifica que os dois centróides auxiliares a_l e a_t, sendo $l \neq t$, devem representar um mesmo grupo;
- **Restrição *cannot-link* em nível de protótipo:** simbolizada por $r'_{cl}(a_l, a_t)$, especifica que os dois centróides auxiliares a_l e a_t, sendo $l \neq t$, devem representar grupos distintos;

No decorrer desta seção os novos tipos de restrições serão detalhados, como segue. O processo de criação das restrições começa criando as restrições *cannot-link* entre representantes (Seção 4.2.1), as quais também auxiliarão a criação das restrições *must-link* entre representantes (Seção 4.2.2). É importante destacar que a novas restrições geradas serão armazenadas em uma estrutura de árvore binária, dessa forma, agilizando os vários acessos a essas restrições durante o processo de definição das mesmas e durante o processo de agrupamento.

4.2.1 Restrição *cannot-link* em Nível de Protótipo

O processo de definição do conjunto de restrições *cannot-link* entre representantes auxiliares é ilustrado pela Figura 4.19. A ideia geral desse processo é encontrar, por meio das instâncias de restrições *cannot-link* R_{cl}, os centróides auxiliares que não deveriam representar um mesmo grupo no processo de detecção de agrupamentos. Para isso, utiliza-se de informações sobre instâncias que participam de alguma restrição *cannot-link*

e de informação sobre os próprios centróides auxiliares definidos anteriormente (Figura 4.19(a)).

Esse processo funciona da seguinte forma, considerando uma restrição $r_{cl}\left(x_i, x_j\right)$, ao observar que um centróide auxiliar a_l está próximo da instância x_i e que um outro centróide auxiliar a_t está próximo da instância x_j, supõe-se que estes dois centróides, a_l e a_t, não devem representar um mesmo grupo, logo, cria-se uma restrição *cannot-link* r'_{ml} entre eles. A Figura 4.19(b) ilustra as restrições *cannot-link* geradas com esse processo.

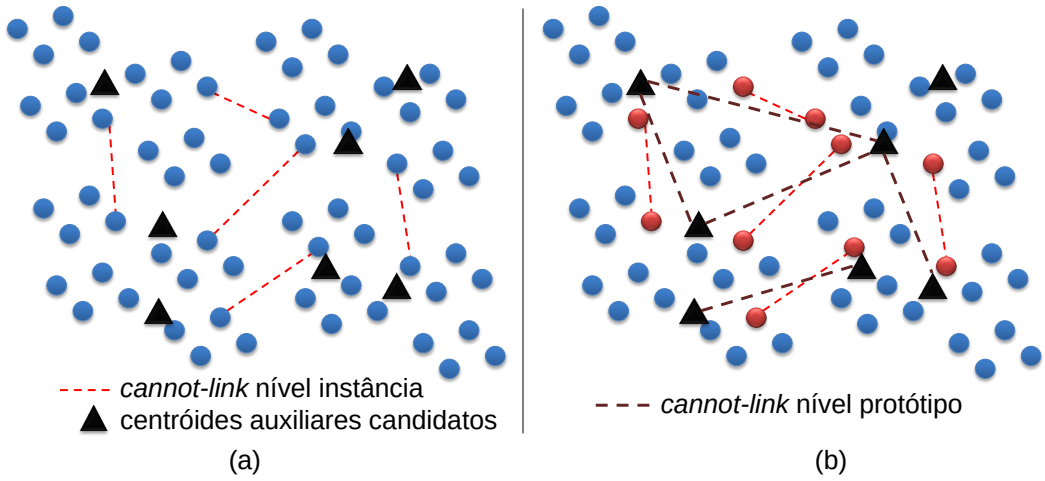

Figura 4.19 – Exemplo do processo de definição das restrições *cannot-link* em nível de protótipo. (a) Conjunto de instâncias com restrições *cannot-link* R_{cl} e conjunto de centróides auxiliares. (b) Restrições *cannot-link* em nível de protótipo definidas.

Com o intuito de esclarecer o funcionamento desse processo, o Algoritmo 5 apresenta os detalhes inerentes a definição das novas restrições *cannot-link*. Como entrada, o algoritmo recebe o conjunto de centróides auxiliares candidatos M_a e um conjunto de restrições *cannot-link* em nível de instância R_{cl}. O processo retorna o conjunto de restrições *cannot-link* em nível de protótipo R'_{cl}.

Algoritmo 5 - `CannotLinkNivelPrototipo`

Entrada: M_a, R_{cl}
Saída: R'_{cl}

1: $R'_{cl} \leftarrow \emptyset$
2: **para** cada restrição $r_{cl}\left(x_i, x_j\right) \in R_{cl}$ **faça**
3: $\quad a_1 = \min_{a_t \in M_a} \delta\left(a_t, x_i\right)$
4: $\quad a_2 = \min_{a_t \in M_a} \delta\left(a_t, x_j\right)$
5: $\quad$ **se** $[a_1 \neq a_2] \wedge [\nexists r'_{cl}\left(a_1, a_2\right) \in R'_{cl}]$ **então**
6: $\quad\quad R'_{cl} \leftarrow R'_{cl} \cup r'_{cl}\left(a_1, a_2\right)$
7: $\quad$ **fim se**
8: **fim para**

O Algoritmo 5 começa criando um conjunto vazio R'_{cl} (linha 1), o qual irá armazenar as novas restrições geradas. Na única estrutura de repetição do algoritmo (linhas

2 a 8) é verificado se as instâncias de cada restrição *cannot-link* $r_{cl} \in R_{cl}$ podem gerar novas restrições *cannot-link* em nível de protótipo. Para isso, (linha 3) busca-se pelo primeiro centróide auxiliar a_1 que esteja mais próximo da primeira instância x_i da restrição $r_{cl}(x_i, x_j)$. Logo após, (linha 4) busca-se pelo segundo centróide auxiliar a_2 que esteja mais próximo da segunda instância x_j da restrição $r_{cl}(x_i, x_j)$. Com esses dois centróides auxiliares, o primeiro passo é verificar se são centróides distintos, em caso afirmativo, então verifica se não existe a restrição com os centróides a_1 e a_2 no conjunto R'_{cl} (linha 5), pois neste caso não haveria a necessidade de criar essa restrição. Após verificar as condições anteriores, a restrição *cannot-link* $r'_{cl}(a_1, a_2)$ finalmente é criada e armazenada no conjunto R'_{cl}. O processo é repetido para todas as restrições do conjunto R_{cl}.

4.2.2 Restrição *must-link* em Nível de Protótipo

O processo de definição do conjunto de restrições *must-link* entre representantes auxiliares é ilustrado pela Figura 4.20. A ideia geral desse processo é unir um centróide auxiliar a outro que esteja mais próximo a ele, desde que, essa união não gere conflito com nenhuma restrição *cannot-link* r'_{cl} criada anteriormente. Para isso, utiliza-se das informações de centróides auxiliares que participam de alguma restrição *cannot-link* (Figura 4.20(a)).

Considerando um conjunto de representantes auxiliares M_a, para cada centróide auxiliar a_i que participa de alguma restrição *cannot-link* r'_{cl}, busca-se por outro centróide auxiliar a_j, que também participe de alguma restrição *cannot-link* r'_{cl}, mais próximo. Participar de alguma restrição *cannot-link* é um requisito necessário, pois assim, tem-se a ideia de que dois centróides mais próximos que não possuem nenhuma restrição *cannot-link* entre si provavelmente devem representar um mesmo grupo. Então, verificando que não haja nenhuma restrição que impeça os centróides a_i e a_j de representarem um mesmo grupo, uma restrição *must-link* é criada entre eles. A Figura 4.20(b) mostra um exemplo da definição de restrições *must-link* em nível de protótipo, salientando que, o centróide no canto superior direito desta figura não tem nenhuma restrição atribuída a ele por não participar de nenhuma restrição *cannot-link* em nível de protótipo.

Com o intuito de esclarecer o funcionamento desse processo, o Algoritmo 6 apresenta os detalhes inerentes a definição das novas restrições *must-link*. Como entrada, o algoritmo recebe o conjunto de centróides auxiliares candidatos M_a e um conjunto de restrições *cannot-link* em nível de protótipo R'_{cl}. O processo retorna o conjunto de restrições *must-link* em nível de protótipo R'_{ml}.

O Algoritmo 6 começa criando um conjunto vazio R'_{ml} (linha 1), o qual irá armazenar as novas restrições geradas. Na única estrutura de repetição do algoritmo (linhas 2 a 8) é selecionado cada centróide auxiliar a_i pertencente a alguma restrição *cannot-link* $r'_{cl} \in R'_{cl}$, onde será verificada a possibilidade da criação de uma restrição *must-link* com outro centróide do conjunto M_a. Para isso, (linha 3) busca-se pelo centróide auxiliar a_j

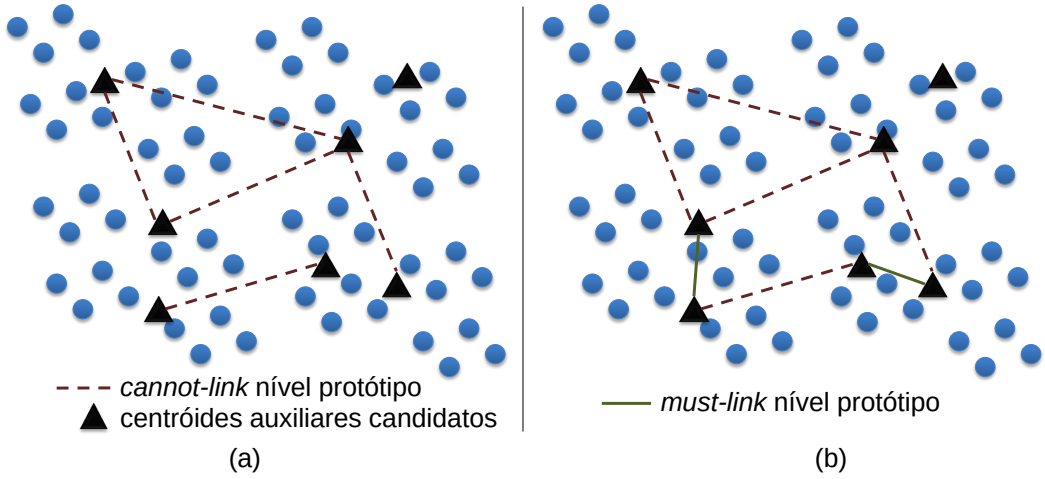

Figura 4.20 – Exemplo do processo de definição das restrições *must-link* em nível de protótipo. (a) Conjunto de instâncias com restrições *cannot-link* R'_{cl} e conjunto de centróides auxiliares. (b) Restrições *must-link* em nível de protótipo definidas.

Algoritmo 6 - `MustLinkNivelPrototipo`

Entrada: M_a, R'_{cl}
Saída: R'_{ml}

1: $R'_{ml} \leftarrow \emptyset$
2: **para** cada $a_i \in M_a$ tal que $\exists r'_{cl}(a_i, *) \in R'_{cl}$ **faça**
3: $a_j = \min_{a_t \in M_a} \delta(a_t, a_i)$
4: **se** $[\nexists r'_{cl}(a_i, a_j) \in R'_{cl}] \wedge [\nexists r'_{ml}(a_i, a_j) \in R'_{ml}]$ **então**
5: $R'_{ml} \leftarrow R'_{ml} \cup r'_{ml}(a_i, a_j)$
6: Aplica função de **transitividade**
7: **fim se**
8: **fim para**

que esteja imediatamente mais próximo do centróide auxiliar a_i, desde que a_j também participe de alguma restrição *cannot-link*. Logo após, (linha 4) é verificado se existe alguma restrição *cannot-link* entre a_i e a_j, caso não exista essa restrição, verifica se já existe uma restrição *must-link* entre esse dois centróides auxiliares no conjunto R'_{ml}. Após verificar as condições anteriores, (linha 5) a restrição *must-link* é gerada e armazenada no conjunto R'_{ml}. A cada geração de uma nova restrição, uma função de transitividade deve ser aplicada por questões de consistência na geração das próximas restrições (linha 6). Essa função será apresentada mais adiante nesta seção. O processo é repetido para todos os centróides auxiliares do conjunto M_a.

Dentro do processo de geração de restrições *must-link* em nível de protótipo existe uma etapa muito importante para manter a consistência entre as restrições *must-link* e *cannot-link*. Nesta etapa, algumas restrições adicionais podem ser geradas para que não haja conflito na geração das próximas restrições durante o processo, e isto é feito considerando o conceito de **transitividade**. A transitividade pode gerar outras restrições

em duas diferentes situações, como apresentado a seguir:

- Se uma nova restrição *must-link* $r'_{ml}(a_i, a_j)$ for gerada e se existir uma restrição *must-link* $r'_{ml}(a_j, a_k) \in R'_{ml}$, pela transitividade pode-se gerar a restrição $r'_{ml}(a_i, a_k)$;
- Se uma nova restrição *must-link* $r'_{ml}(a_i, a_j)$ for gerada e se existir uma restrição *cannot-link* $r'_{cl}(a_j, a_k) \in R'_{cl}$, pela transitividade pode-se gerar a restrição $r'_{cl}(a_i, a_k)$.

A complexidade computacional dos Algoritmos 5 e 6 é discutida a seguir. O Algoritmo 5, tem um laço de repetição que será executado t vezes, em que $t = |R_{cl}|$ (linha 2). Dentro desse laço são executadas duas varreduras do conjunto de auxiliares M_a de tamanho $2l$, em que $l = |M_a|$ (linhas 3 e 4), e uma busca em uma árvore binária R'_{cl} que custa $\log c$ no pior caso, sendo $c = |R'_{cl}|$ (linha 5). Dessa forma, o Algoritmo 5 tem o custo $t * (O(l) + O(\log c))$ resultando em $O(t * (l + \log c))$. Já o Algoritmo 6, tem um laço de repetição que será executado no máximo l vezes, novamente considerando $l = |M_a|$ (linha 2). Dentro desse laço é executada uma varredura no conjunto de centróides auxiliares, também de tamanho l (linha 3), e duas buscas em árvores binárias distintas, com custo $\log c$ e $\log m$, sendo $c = |R'_{cl}|$ e $m = |R'_{ml}|$ respectivamente (linha 4). Dessa forma, o Algoritmo 6 tem o custo $l * (O(l) + O(\log c) + O(\log m))$ resultando em $O(l^2 * (\log c + \log m))$. Portanto, o algoritmo para a geração das restrições *cannot-link* tem complexidade linear em função da quantidade de restrições em nível de instância informadas. Já o algoritmo para a geração das restrições *must-link* tem complexidade quadrática no tamanho do conjunto de centróides auxiliares M_a, considerando o caso em que todos os centróides auxiliares participam de alguma restrição *cannot-link*.

4.3 Diversidade

Dependendo da cardinalidade de R_{ml}, um grande número de centróides auxiliares pode ser gerado. O problema relacionado a essa excessiva quantidade de representantes auxiliares disponíveis para cada grupo é que nem todos esses centróides contribuem para a detecção das formas arbitrárias que os grupos possuem, sendo um dos motivos a proximidade entre esses representantes auxiliares na extensão dos grupos. Além disso, a cardinalidade de M_a tem uma influência direta sobre o número de cálculos de distância necessários (apresentado mais adiante na Seção 4.4.1). Assim, é importante selecionar somente os centróides auxiliares essenciais em M_a para que o processo de agrupamento funcione de maneira eficiente. Dessa forma, a ideia é selecionar os centróides auxiliares mais distantes do centróide principal e mais distantes entre os próprios representantes auxiliares.

Na política adotada pelo MRS-*kmeans* para a seleção dos centróides auxiliares mais informativos, assume-se que se quer encontrar pares de centróides auxiliares mais diversos

em M_a atribuídos a cada grupo C_p. Os passos executados pelo algoritmo MRS-*kmeans* para realizar essa seleção foram inspirados pela estratégia adotada em (GUHA; RASTOGI; SHIM, 1998; PATERLINI; NASCIMENTO; JR., 2011) e são descritos a seguir. Para exemplificar, considere a Figura 4.21, a qual representa um conjunto de centróides auxiliares candidatos, derivados de restrições *must-link* entre instâncias r_{ml}, e distribuídos por toda a extensão da estrutura de um grupo (Figura 4.21(a)) em um conjunto de dados.

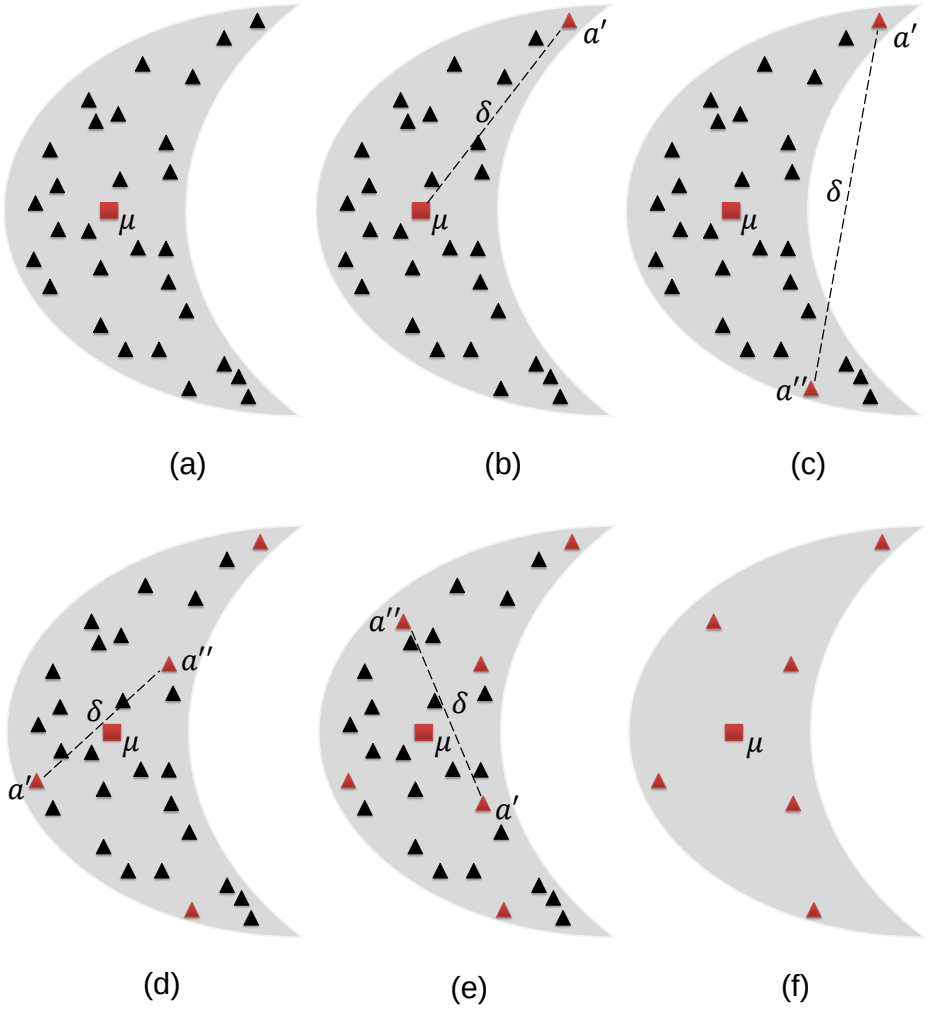

Figura 4.21 – Exemplo de execução do algoritmo de diversidade, em que o centróide principal é representado pelo quadrado e os centróides auxiliares são representados pelos triângulos. (a) Estado inicial. (b - e) Processo de busca por centróides auxiliares diversos. (f) Centróides auxiliares utilizados para representar o grupo.

O primeiro passo desse algoritmo, mostrado na Figura 4.21(b), é buscar por um centróide auxiliar candidato que esteja mais distante do centróide principal μ, sendo o primeiro elemento (a') de um par de centróides auxiliares. Logo após, o algoritmo busca pelo centróide auxiliar candidato (a'') mais distante de a', formando assim o primeiro par de centróides auxiliares diversos (Figura 4.21(c)), ou seja, mais distantes do centro (μ) e mais

distantes entre si. O processo continua buscando por novos pares de centróides auxiliares (Figuras 4.21(d) e 4.21(e)) mais distantes do centróide principal e mais distantes de pares já encontrados anteriormente, até que alcance uma quantidade desejada desses auxiliares. Ao final, como mostra a Figura 4.21(f), um número reduzido de centróides auxiliares diversos serão utilizados durante o processo de agrupamento, diminuindo assim, o custo do cálculo de distância.

A fim de esclarecer o funcionamento desse processo, o Algoritmo 7 apresenta os detalhes inerentes a busca por centróides auxiliares diversos dentro de um grupo. Como entrada, o algoritmo recebe o conjunto de centróides principais M_μ, o conjunto de centróides auxiliares candidatos M_a e uma porcentagem Z de auxiliares desejados para compor cada grupo. A partir disso, o algoritmo retorna um conjunto de centróides auxiliares que irão contribuir para o processo de agrupamento.

Algoritmo 7 - `Diversidade`

Entrada: M_μ, M_a, Z
Saída: M'_a

1: **para** $S_i \subseteq M_a$ tal que S_i representa o grupo com centróide $\mu_i \in M_\mu$ **faça**
2: $S'_i \leftarrow \emptyset$
3: $S'_i \leftarrow a'_1 \in S_i$ mais distante de μ_i
4: $S'_i \leftarrow a''_1 \in S_i$ mais distante de a'_1 e μ_i
5: $\sigma \leftarrow \delta\left(a'_1, a''_1\right)$
6: **para** $g \leftarrow 2$ até $|S'_i| \leq \left\lceil \frac{Z*|M_a|}{100} \right\rceil$ **faça**
7: $a'_g = \min_{j=1}^{n} \sum_{t=1}^{g-1} \{|\sigma - \delta\left(a'_t, a_j\right)| + |\sigma - \delta\left(a''_t, a_j\right)|\}$
8: $a''_g = \min_{j=1}^{n} \sum_{t=1}^{g-1} \{|\sigma - \delta\left(a'_t, a_j\right)| + |\sigma - \delta\left(a''_t, a_j\right)|\} + \left|\sigma - \delta\left(a'_g, a_j\right)\right|$
9: $S'_i \leftarrow S'_i \cup \left\{a'_g, a''_g\right\}$
10: $\sigma \leftarrow \sigma + \delta\left(a'_g, a''_g\right)$
11: **fim para**
12: $M'_a \leftarrow S'_i$
13: **fim para**

Na primeira estrutura de repetição do Algoritmo 7 (linhas 1 a 13), é selecionado um dos k subconjuntos de centróides auxiliares candidatos $S_i \subseteq M_a$ que foram anteriormente atribuídos aos respectivos grupos cujo centróide principal $\mu_i \in M_\mu$ está mais próximo. Um novo subconjunto S'_i é criado como um conjunto vazio (linha 2), onde, posteriormente é adicionado o primeiro par de centróides auxiliares diversos (linhas 3 e 4), da seguinte forma: primeiramente, busca-se pelo centróide auxiliar a' mais distante do centróide principal μ_i; logo após, busca-se pelo centróide auxiliar a'' mais distante do primeiro centróide auxiliar a' e do centróide principal μ_i. A distância entre esse primeiro par de centróides auxiliares é calculada e armazenada para futuro uso em uma variável σ (linha 5). A segunda estrutura de repetição (linhas 6 a 11) garante a descoberta dos próximos pares de centróides auxiliares diversos até que uma desejada porcentagem Z seja alcançada. O primeiro centróide auxiliar do próximo par a'_g é encontrado pelo cálculo da função

realizada na linha 7 do algoritmo. Essa função é usada para calcular a distância de um centróide auxiliar do subconjunto S_i com os pares já encontrados anteriormente (a' e a'') e o somatório das distâncias entre os pares anteriores σ, retornando o centróide auxiliar que minimiza essa função. Para o segundo centróide auxiliar do par a''_g calcula-se as mesmas distâncias anteriores, mais a distância para o primeiro auxiliar desse par atual a'_g (linha 8), novamente retornando o centróide auxiliar que minimiza a respectiva função. O novo par de centróides é adicionado ao subconjunto de centróides auxiliares S_i e a distância entre eles é incrementada na variável σ (linhas 9 e 10, respectivamente). No momento em que for alcançada a porcentagem Z, que representa a quantidade de auxiliares proporcional a quantidade de centróides em M_a, o conjunto de centróides auxiliares diversos de um determinado grupo terá sido encontrado. O processo então, passa para a seleção dos centróides auxiliares candidatos de um próximo grupo, até que todos os grupos tenham sido executados.

Em relação a complexidade computacional do Algoritmo 7, a primeira estrutura de repetição será executa k vezes, referente aos subconjuntos de centróides auxiliares candidatos $S_i \subseteq M_a$ que foram atribuídos aos k diferentes grupos. A primeira etapa (linhas 3 e 4) realiza duas varreduras no subconjunto de centróides auxiliares candidatos S_i, comparando cada centróide aos elementos do primeiro par de centróides auxiliares diversos (a' e a''), logo, $2\,|S_i|$ execuções, resultando em complexidade $O\,(|S_i|)$. Em cada uma das k iterações do algoritmo são executadas g varreduras para cada um dos elementos dos pares subsequentes, com complexidade $O\,(|S_i|)$. Assim, pode-se definir a complexidade do uso desse algoritmo no processo de agrupamento em $O\,(k * (|S_i| + (g * |S_i|)))$. Dessa forma, tem-se um custo linear em função da quantidade de centróides auxiliares candidatos $S_i \subseteq M_a$ de cada grupo C_i. Um problema observado ao uso de qualquer algoritmo para realizar essa tarefa estaria relacionado ao seu custo computacional, pois, centróides auxiliares são selecionados, dentre os candidatos, em cada iteração do MRS-*kmeans* (mais detalhes na Seção 4.4.2). Portanto, o baixo custo computacional de tal algoritmo apresentado torna-o computacionalmente praticável.

4.4 Método de Agrupamento

A partir de um conjunto de dados multidimensionais $X = \{x_1, ..., x_n\}$, um conjunto de restrições *must-link* em nível de instância R_{ml}, um conjunto de restrições *cannot-link* em nível de instância R_{cl} e um número k de grupos, o algoritmo MRS-*kmeans* retorna k partições dos dados em X. No processamento do algoritmo, as restrições entre instâncias, informadas externamente, não são utilizadas dentro do processo de agrupamento propriamente dito, mas sim, para gerar as informações necessárias para que o MRS-*kmeans* possa funcionar. Nesse caso, é necessária uma fase anterior a fase de detecção dos agrupamentos, a qual será denominada: pré-processamento.

No pré-processamento, as restrições em nível de instância são trabalhas para gerar os centróides auxiliares candidatos, as restrições *cannot-link* em nível de protótipo e as restrições *must-link* em nível de protótipo, como mostra o Algoritmo 8. É importante salientar que, a fase de pré-processamento não precisa ser necessariamente executada toda vez que for preciso gerar algum agrupamento em um conjunto de dados. As informações de centróides auxiliares candidatos e restrições em nível de protótipo podem ser armazenadas e depois utilizadas no processo de agrupamento de fato. Em um sistema de mineração de dados, uma boa prática seria armazenar essas informações como uma espécie de metadados para cada respectivo conjunto de dados.

Algoritmo 8 - `Pré-Processamento`

Entrada: R_{ml}, R_{cl}
Saída: M_a, R'_{ml}, R'_{cl}
1: $M_a \leftarrow$ `RepresentantesAuxiliaresCandidatos` (R_{ml})
2: $R'_{cl} \leftarrow$ `CannotLinkNivelProtótipo` (M_a, R_{cl})
3: $R'_{ml} \leftarrow$ `MustLinkNivelProtótipo` (M_a, R'_{cl})

Após a etapa de pré-processamento, tem-se os dados necessários para gerar o particionamento de um conjunto de dados. Mas antes de apresentar o funcionamento desse processo, faz-se necessário entender como utilizar os múltiplos representantes dentro de um processo de agrupamento. A função de distância agregada é um ponto importante durante todo esse processo e será detalhada na Seção 4.4.1. Logo adiante, na Seção 4.4.2 será apresentado o fechamento de todo o processo do algoritmo MRS-*kmeans*.

4.4.1 Função de Distância Agregada

A fim de considerar ambos tipos de centróides, principal e auxiliar, no cálculo da similaridade entre instâncias e representantes, realizado pelo algoritmo, é necessário definir uma função de distância agregada que possa tirar maior vantagem da informação fornecida por todos os representantes de grupos. Quando atribui-se instâncias aos grupos, cada instância $x_i \in X$ é avaliada pela função de distância agregada δ_g definida pela Equação 4.21, como segue.

$$\delta_g\left(Q_k, x_i\right) = \min_{q_j \in Q_k} \left(\delta\left(q_j, x_i\right) \cdot \frac{1}{w_j} \right) \tag{4.21}$$

Nesta equação, Q_k é o conjunto de representantes de um grupo C_k (principal e auxiliares), $\delta\left(\right)$ é uma função de distância e w_j é o peso do representante q_j. Esta função de distância agregada é um caso específico de uma função genérica apresentada no Capítulo 2. A semântica de sua aplicação está relacionada a tentativa de obter instâncias mais próximas de qualquer centróide do grupo, permitindo capturar as várias distribuições espaciais.

O peso dado a cada representante é de suma importância para o processo. Sendo o centróide principal o elemento mais representativo do grupo atribui-se $w_\mu \leq 1$. Para os representantes auxiliares são atribuídos pesos variáveis, considerando $w_a > 1$. Como cada componente conexa, definida por uma ou mais restrições, pode ter um diferente número de instâncias interconectadas, logo, o peso w_i definido para cada centróide auxiliar $a_i \in M_a$ é proporcional ao número de instâncias que compõem a componente conexa. A ideia é que centróides auxiliares que representam mais instâncias devem ter mais influencia ao representar um grupo. Portanto, centróides principais terão pesos semelhantes entre si e superiores aos pesos dos centróides auxiliares, os quais podem ter pesos diversos dependendo da cardinalidade de cada componente conexa.

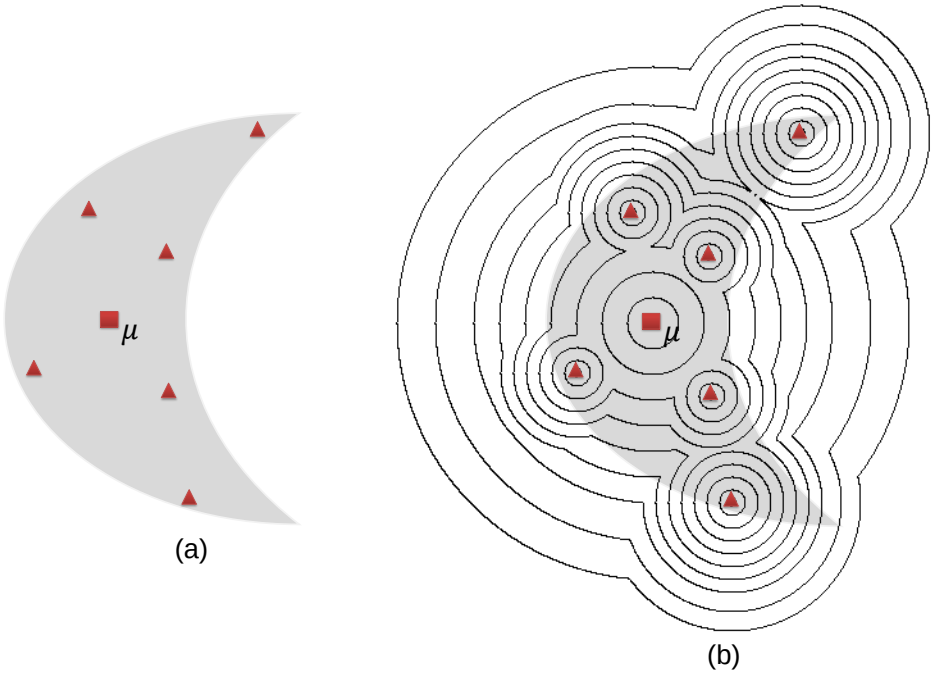

Figura 4.22 – Exemplo de atuação da função de distância agregada. (a) Conjunto de representantes em um grupo. (b) Abrangência de cada representante considerando seu respectivo peso.

Para exemplificar o comportamento dessa função de distância, a Figura 4.22 apresenta, em destaque, um grupo de um conjunto de dados. O grupo é representado por diversos centróides auxiliares, em forma de triângulos, e por seu centróide principal μ, como mostra a Figura 4.22(a). Durante a fase de atribuição das instâncias aos grupos, a distância de cada instância para cada representante será calculada. No caso desta função de distância agregada, a instância será atribuída ao representante que cobre a região em que essa instância está alocada. A Figura 4.22(b) mostra a área de abrangência de cada representante sobre o conjunto. É importante notar que, como o centróide principal tem maior peso, sua área de atuação será maior que a dos demais representantes.

4.4.2 Agrupamento

O processo completo realizado pelo algoritmo MRS-*kmeans* é descrito nesta seção. Até o momento já se pode processar toda informação necessária, como geração dos centróides auxiliares e restrições em nível de protótipo, para realizar a abordagem de detecção de agrupamento proposta nesta dissertação. A Figura 4.23 traz um fluxograma apresentando todas as etapas envolvidas no processamento do MRS-*kmeans*. Percebe-se dessa forma, que o algoritmo MRS-*kmeans* é bastante semelhante ao clássico algoritmo *k-means*.

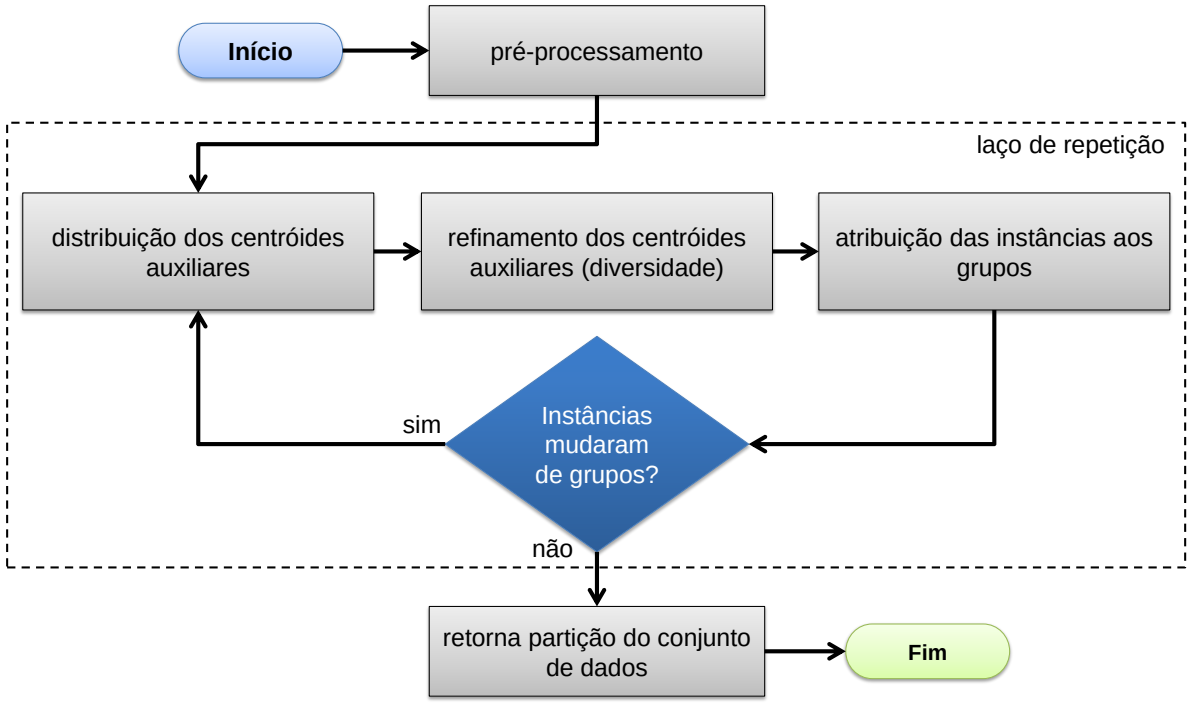

Figura 4.23 – Fluxograma do processo de agrupamento realizado pelo MRS-*kmeans*.

Como mostra a Figura 4.23, o primeiro passo é realizar a fase de pré-processamento (ilustrada no Algoritmo 8), onde são extraídas as informações principais para o funcionamento do algoritmo. Logo em seguida está a fase de distribuição dos centróides auxiliares aos grupos. Por questões de otimização, uma fase de refinamento se faz necessária para diminuir a quantidade de centróides auxiliares por grupo, selecionando apenas aqueles que mais contribuem para a detecção da real estrutura do agrupamento. Na próxima fase todas as instâncias são atribuídas aos seus respectivos grupos, com menor distância de seus representantes. O processo verifica se houve alguma alteração nos grupos de uma iteração para outra e decide se deve realizar mais uma iteração, ou se finaliza e retorna o particionamento do conjunto de dados.

A fim de apresentar os detalhes desse processo, o Algoritmo 9 mostra como serão utilizados todos os procedimentos descritos neste capítulo para gerar o particionamento de um conjunto de dados. Como entrada, o algoritmo recebe um conjunto de dados X, um

número de grupos k, a porcentagem Z que deve ser selecionada de centróides auxiliares, além dos dados gerados na fase de pré-processamento. Esses dados compreendem: o conjunto de candidatos a representantes auxiliares M_a e as restrições em nível de protótipo R'_{ml} e R'_{cl}. A partir disso, o algoritmo retorna um particionamento dos dados Π.

Algoritmo 9 - `MRS-kmeans`

Entrada: X, M_a, R'_{ml}, R'_{cl}, k, Z
Saída: $\Pi = \{C_1, ..., C_k\}$

1: Seleciona aleatoriamente k centróides principais iniciais $M_\mu = \{\mu_1, ..., \mu_k\}$
2: **repita**
3: $M'_a \leftarrow \emptyset$
4: /* atribui cada centróide auxiliar ao grupo mais próximo */
5: **para** cada auxiliar $a_i \in M_a$ **faça**
6: $\mu_{mp} = \min_{j=1}^{k} \delta(a_i, \mu_j)$
7: **se** `VerificaRestrições` $(a_i, \mu_{mp}, R'_{ml}, R'_{cl})$ **então**
8: $a_i.representa \leftarrow \mu_{mp}$
9: **fim se**
10: **fim para**
11: /* seleciona centróides auxiliares mais diversos */
12: $M'_a \leftarrow$ `Diversidade` (M_a, M_μ, Z)
13: **para** cada grupo $C_j \in \Pi$ **faça**
14: $Q_j \leftarrow \mu_j \cup S_j \in M'_a$
15: **fim para**
16: /* atribui cada instância ao grupo mais próximo */
17: **para** cada instância $x_i \in X$ **faça**
18: $\mu_{mp} = \min_{j=1}^{k} \delta_g(Q_j, x_i)$
19: $C_{mp} \leftarrow C_{mp} \cup x_i$
20: **fim para**
21: /* atualiza os centróides principais */
22: **para** cada grupo $C_j \in \Pi$ **faça**
23: $\mu_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$
24: **fim para**
25: **até** convergência pela Equação 4.22
26: **retorna** $\Pi = \{C_1, ..., C_k\}$

O Algoritmo 9 inicia selecionando aleatoriamente k instâncias do conjunto de dados como centróides principais iniciais (linha 1). Uma estrutura de repetição global (linhas 2 a 25) executa cada iteração do processo até que um critério de parada exigido seja atingido, tal critério de parada será apresentado mais adiante nesta seção. Um conjunto de representantes auxiliares M'_a é inicializado como um conjunto vazio (linha 3). Pode-se notar que a cada iteração o conjunto M'_a será declarado como vazio, e isso é um comportamento esperado para o algoritmo, pois, a cada iteração novos centróides auxiliares diversos deverão ser buscados devido principalmente a movimentação do centróide principal. Na estrutura de repetição das linhas 5 a 10, cada centróide auxiliar será atribuído ao grupo do qual o centróide principal está mais próximo, verificando sempre a não vi-

olação das restrições em nível de protótipo. Em seguida, são selecionados os centróides auxiliares mais diversos de cada grupo (linha 12), os quais representarão seus respectivos grupos posteriormente (linhas 13 a 15). O próximo passo do algoritmo é atribuir todas as instâncias a seus respectivos grupos (linhas 17 a 20). Nesse passo, a função de distância agregada é considerada para representar a distância de um grupo a uma instância. Na sequencia, os centróides principais são atualizados com base nas instâncias que foram atribuídas a cada grupo, e é verificado se o algoritmo já atingiu a convergência. Caso não tenha atingido a convergência, o algoritmo retorna para linha 3 e realiza outra iteração.

O critério de parada do algoritmo MRS-*kmeans*, da mesma forma que o algoritmo *k-means*, é dado pelo cálculo do erro quadrático, mostrado na Equação 4.22. Dessa forma, o algoritmo só pode parar se $|E^t - E^{t-1}| \leq \epsilon$, para um dado limite $\epsilon > 0$, sendo t a iteração atual. Isto é, ele só deve parar se a diferença entre o erro quadrático da iteração atual e anterior for menor ou igual a um dado limite ϵ, que geralmente é um valor bem próximo de zero.

$$E = \sum_{j=1}^{k} \sum_{x_i \in C_j} \delta_g \left(Q_j, x_i\right)^2 \tag{4.22}$$

A verificação das restrições, mostrada no Algoritmo 10, funciona de forma um pouco diferente da verificação proposta no algoritmo COP-*kmeans* (apresentado no Capítulo 3). O Algoritmo 10 recebe como entrada um centróide auxiliar a_i, um centróide principal μ_{mp} que representa o grupo mais próximo de a_i, e os dois conjuntos de restrições em nível de protótipo R'_{ml} e R'_{cl}. No COP-*kmeans* as restrições eram verificadas somente para dizer se era possível ou não atribuir uma instância a um grupo. Já na verificação de restrições descrita aqui, o algoritmo primeiramente verifica a possibilidade de atribuição de um centróide auxiliar a_i a um grupo C_{mp} pelas restrições *cannot-link* R'_{cl} (linhas 1 a 5). Caso não haja nenhuma restrição que impeça essa atribuição, todo centróide auxiliar a_j que compartilhe uma restrição *must-link* R'_{ml} com o centróide a_i será, assim como ele, atribuído ao grupo mais próximo C_{mp} (linhas 6 a 8).

Algoritmo 10 - `VerificaRestrições`

Entrada: a_i, μ_{mp}, R'_{ml}, R'_{cl}
Saída: **true** ou **false**

1: **para** cada restrição $r'_{cl}\left(a_i, a_j\right) \in R'_{cl}$ **faça**
2: **se** $a_j.representa = \mu_{mp}$ **então**
3: **retorna false**
4: **fim se**
5: **fim para**
6: **para** cada restrição $r'_{ml}\left(a_i, a_j\right) \in R'_{ml}$ **faça**
7: $a_j.representa \leftarrow \mu_{mp}$
8: **fim para**
9: **retorna true**

Em relação a complexidade computacional do Algoritmo 10, a primeira etapa verifica todas as restrições *cannot-link* do conjunto R'_{cl}, sendo um conjunto de tamanho $c = |R'_{cl}|$, resultando assim, em complexidade $O(c)$. De forma semelhante, a segunda etapa desse algoritmo de verificação realiza uma varredura em todas as restrições *must-link* do conjunto R'_{ml}, sendo um conjunto de tamanho $m = |R'_{ml}|$, resultando em complexidade $O(m)$. Desta forma, pode-se definir o custo computacional do Algoritmo 10 em $O(c + m)$. Na próxima seção será apresentada a complexidade geral do algoritmo MRS-*kmeans*, destacando separadamente a fase de pré-processamento e a fase de geração de particionamento do conjunto de dados.

4.5 Complexidade

O processo geral do algoritmo MRS-*kmeans* pode ser dividido em duas partes principais, a fase de pré-processamento e a fase de detecção dos agrupamentos. Dentre as funções realizadas na fase de pré-processamento, apresentadas pelo Algoritmo 8, estão: a definição dos centróides auxiliares, com custo $O(|R_{ml}| + |I_m|)$; a definição das restrições *cannot-link* em nível de protótipo, com custo $O(|R_{cl}| * (|M_a| + \log |R'_{cl}|))$; e a definição das restrições *must-link* em nível de protótipo, com custo $O\left(|M_a|^2 * (\log |R'_{cl}| + \log |R'_{ml}|)\right)$. Portanto, a fase de pré-processamento possui um custo quadrático em função do tamanho do conjunto de centróides auxiliares candidatos M_a, que por sua vez é diretamente influenciado pela quantidade de restrições em nível de instância informadas ao MRS-*kmeans*.

Em relação a complexidade computacional do processo de agrupamento, mostrado pelo Algoritmo 9, a fase de atribuição dos centróides aos respectivos grupos (linhas 5 a 10) possui uma estrutura de repetição que será executada $|M_a|$ vezes. Dentro dessa estrutura tem-se uma varredura aos k grupos e uma verificação das restrições que, como já visto anteriormente, tem custo $O(|R'_{cl}| + |R'_{ml}|)$. Resultando no custo computacional $O(|M_a| * (k + |R'_{cl}| + |R'_{ml}|))$. A próxima fase, que realiza a seleção dos centróides auxiliares candidatos mais diversos (linhas 12 a 15), como mostrado anteriormente, tem o custo computacional $O(k * (|S_i| + (g * |S_i|)))$. Já a fase de atribuição das instâncias ao grupo mais próximo (linhas 17 a 20), tem uma estrutura de repetição que será executada n vezes, sendo n o tamanho do conjunto de dados X. Nesta fase, para cada instância do conjunto de dados será feita uma varredura nos k grupos de m representantes. Para que o custo dessa etapa não fosse muito elevado, a distância entre os centróides auxiliares e as instâncias, são armazenadas em uma matriz de distâncias, pois essas distâncias não se alteram. Com isso, o custo do cálculo de distância das instâncias para os m centróides auxiliares serão calculados somente uma vez. Resultando assim na complexidade $O(n * k)$. A fase de atualização dos centróides principais (linhas 22 a 24) tem custo de uma varredura no conjunto de dados X, portanto, sua complexidade é $O(n)$. Por fim, o algoritmo MRS-*kmeans* pode repetir os passos descritos acima em até t iterações. Isso faz com que

a complexidade total do algoritmo seja linear em função principalmente da quantidade de instâncias, da quantidade de iterações necessárias e da quantidade de centróides auxiliares candidatos, isto é, complexidade $O\left(t * (n + |M_a|)\right)$.

4.6 Considerações Finais

Este capítulo apresentou todos os detalhes referentes ao algoritmo MRS-*kmeans*, proposto para realizar agrupamento de dados por meio de semi-supervisão. O novo método consiste em extrair informações de semi-supervisão para gerar múltiplos representantes auxiliares e incorporá-los ao processo de particionamento de dados baseado no clássico algoritmo *k-means*. Um ponto fundamental da fase de extração de informações é a geração de restrições entre os novos representantes criados. Essas restrições atuam informando se um respectivo par de representantes auxiliares deve ou não auxiliar um determinado grupo dentro do conjunto de dados. Outra questão importante é o uso da diversidade na seleção de protótipos mais representativos de cada grupo, contribuindo assim, para reduzir o custo gerado com o cálculo da distância agregada. O método proposto tem maior eficácia na detecção de agrupamentos de formas arbitrárias, quando comparado a outros algoritmos de semi-supervisão que utilizam uma abordagem de particionamento de dados. No próximo capítulo são apresentados os experimentos realizados e a discussão sobre os resultados obtidos.

Capítulo 5

Experimentos e Análise dos Resultados

Neste capítulo são apresentados e discutidos os diversos experimentos realizados que ajudam a corroborar a afirmação de que o algoritmo MRS-*kmeans* é superior a algoritmos da literatura, utilizando a abordagem de particionamento de dados ao determinar agrupamentos em conjuntos de dados de formas diversas. O capítulo está organizado como segue. A Seção 5.1 apresenta os métodos utilizados para a realização dos experimentos. Na seção 5.2 são apresentados os resultados e discussões sobre os experimentos realizados. Por fim, a Seção 5.3 traz as considerações finais deste capítulo.

5.1 Método de Avaliação

Os experimentos foram realizados em um *desktop* Dell XPS-8700 com processador Intel QuadCore i5-4430 CPU@3.00GHz, 8GB de memória RAM e disco rígido SATA-III 1TB 7200 RPM, utilizando o compilador GNU gcc sobre Microsoft Windows 8 64-bits. Para realizar tais experimentos foram utilizados quarenta conjuntos de dados sintéticos e cinco conjuntos de dados reais. Os conjuntos de dados sintéticos foram gerados por meio da ferramenta *Distribution Painter* (DISTRIBUTIONPAINTER, 2014), apresentada no trabalho descrito em (ALBUQUERQUE; LÖWE; MAGNOR, 2011). Com o uso desta ferramenta é possível manter uma estrutura padrão para um dado conjunto de dados mesmo variando a sua dimensionalidade. No trabalho descrito nesta dissertação foram definidas oito estruturas diferentes para a geração dos conjuntos de dados sintéticos. Essas estruturas foram consideradas para a criação de conjuntos, inicialmente, em duas dimensões e que posteriormente foram alterados para a obtenção de conjuntos com a mesma estrutura em espaços de maior dimensionalidade. A ferramenta *Distribution Painter* permite a visualização em três dimensões de qualquer conjunto d-dimensional, sendo $d \geq 3$, selecionando um trio de dimensões por vez, e também permite fazer correlações entre pares de dimensões, possibilitando assim, visualizar o conjunto e estabelecer uma estrutura desejada para os dados em qualquer par de dimensões. A Figura 5.24 ilustra a ferramenta no momento da criação de um dos conjuntos de dados sintéticos em três dimensões.

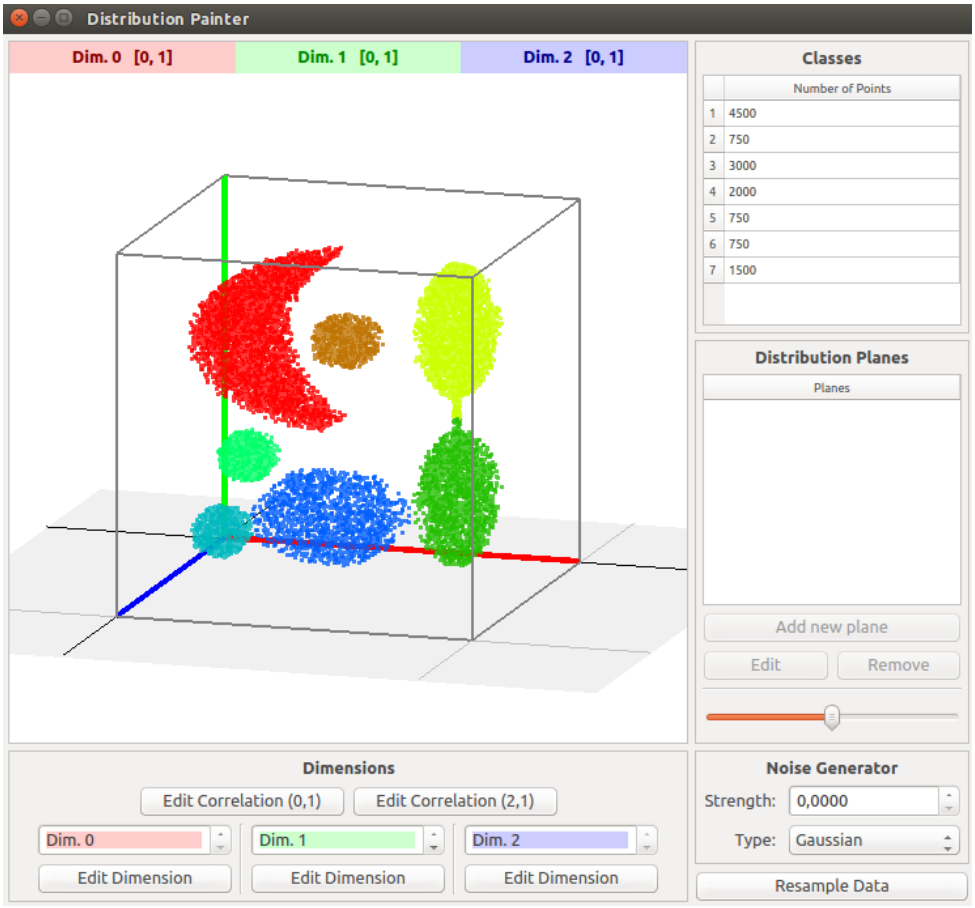

Figura 5.24 – Ilustração da área de trabalho da ferramenta *Distribution Painter*.

A Tabela 5.2 apresenta as informações dos quarenta conjuntos de dados sintéticos gerados para os experimentos, com 2, 3, 4, 6 e 8 dimensões. Para a criação de cada conjunto houve a preocupação em aumentar a quantidade de instâncias a medida que aumentava-se a quantidade de dimensões, pois, ao aumentar o número de dimensões de um determinado conjunto de dados a área de um grupo eventualmente cresce e, neste caso, pode-se obter um conjunto muito esparso. Os conjuntos de dados reais foram utilizados na realização de um estudo de caso descrito a seguir na Seção 5.2.5. Esses conjuntos de dados foram obtidos na base de dados do Instituto de Pesquisa Econômica Aplicada (IPEA, 2015), mais detalhes sobre os dados são apresentados na Seção 5.2.5.

A estrutura dos conjuntos de dados sintéticos em duas dimensões é ilustrada na Figura 5.25. A partir desses conjuntos, em 2D, foram gerados os outros conjuntos com maiores dimensionalidades, ou seja, para cada conjunto de dados a mesma estrutura de grupos é mantida apesar de diferentes números de dimensões. Os diferentes formatos para esses conjuntos foram inspirados pelos trabalhos descritos em (GUHA; RASTOGI; SHIM, 1998; GIONIS; MANNILA; TSAPARAS, 2007), que também possuem o intuito de testar a capacidade de algoritmos em encontrar as formas arbitrárias das estruturas de cada conjunto. No caso desses quarenta conjuntos de dados, houve a preocupação em

Tabela 5.2 – Informações dos conjuntos de dados sintéticos.

Conjuntos de Dados	**#grupos**	**#instâncias**				
		2D	**3D**	**4D**	**6D**	**8D**
DB1	2	4.000	6.000	8.000	12.000	16.000
DB2	3	5.000	7.500	10.000	15.000	20.000
DB3	5	6.000	9.000	12.000	18.000	24.000
DB4	6	3.000	4.500	6.000	9.000	12.000
DB5	3	10.000	15.000	20.000	30.000	40.000
DB6	2	7.000	10.000	14.000	21.000	28.000
DB7	7	9.000	13.250	18.000	27.000	36.000
DB8	4	9.000	13.250	18.000	27.000	36.000

gerar grupos de diferentes tamanhos, formas e densidades, e no caso dos conjuntos DB7 e DB8, diferentes agrupamentos para uma mesma estrutura dos dados. Pode-se perceber na Figura 5.25 as diferentes cores para os grupos em cada conjunto de dados. Essas cores ajudarão a identificar os grupos dos particionamentos gerados pelos algoritmos em uma fase de experimentos apresentada logo a seguir neste capítulo.

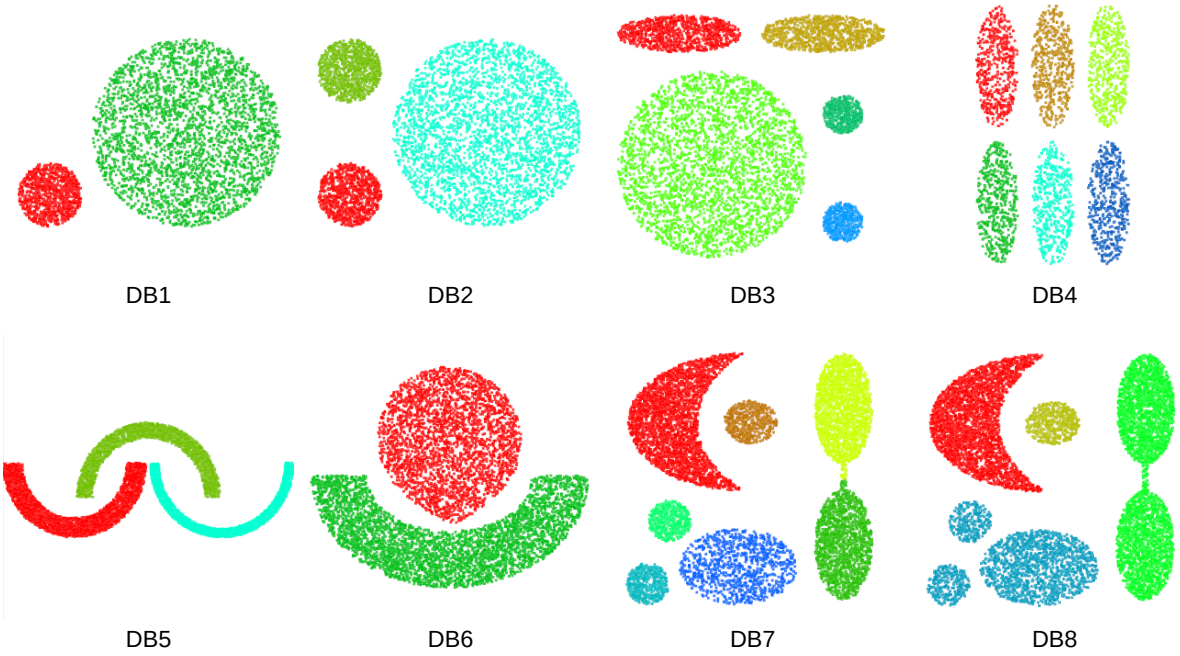

Figura 5.25 – Ilustração dos conjuntos de dados sintéticos. Para cada conjunto, são utilizadas cores diferentes para representar cada classe distinta.

Os experimentos realizados compararam o algoritmo MRS-*kmeans* com dois algoritmos da literatura: o algoritmo COP-*kmeans* (WAGSTAFF et al., 2001) e o algoritmo MLC-*kmeans* (HUANG; CHENG; ZHAO, 2008). Esses algoritmos foram escolhidos para a comparação por serem, assim como o MRS-*kmeans*, variações do clássico algoritmo *k-means* e por possuírem abordagens semelhantes ao algoritmo gerado no trabalho desta dissertação. Ambos algoritmos realizam agrupamento semi-supervisionado e, além disso,

o MLC-*kmeans* emprega um representante auxiliar por grupo. Durante o processo de agrupamento de cada algoritmo a distância Euclidiana foi utilizada no cálculo de distância realizado na fase de atribuição das instâncias aos grupos. Para avaliação de desempenho dos algoritmos foi utilizada a metodologia específica para algoritmos de semi-supervisão, apresentada na Seção 3.3, além do índice *Rand*, muito utilizado em comparações de algoritmos de detecção de agrupamentos, apresentado na Seção 2.1.4.1.

Os conjuntos de restrições foram gerados para cada conjunto de dados a partir de seus rótulos, os quais já são previamente conhecidos. Essas restrições foram geradas da seguinte forma: a cada iteração eram selecionadas duas instâncias aleatoriamente do conjunto de dados, logo em seguida, verificava-se seus rótulos para que, se fossem instâncias do mesmo grupo, gerassem uma restrição *must-link* ou para que, instâncias de grupos distintos, gerassem uma restrição *cannot-link*. Para cada conjunto de dados sintético foram gerados 10 conjuntos de restrições, cada um com restrições em 1%, 2%, 3%, 4%, 5%, 6%, 7%, 8%, 9% e 10% das instâncias do conjunto. Já para o conjunto de dados reais, foi gerado um conjunto de restrições em 10% das instâncias do conjunto. Nos trabalhos descritos na literatura de semi-supervisão, geralmente são utilizadas 10%, 20% ou até 30% de instâncias como fonte de informação adicional para os algoritmos, porém, ao longo do desenvolvimento do trabalho descrito aqui, percebeu-se que quanto menos restrições o algoritmo precisa ter para obter um resultado satisfatório, mais robusto ele será. Por esse motivo, utilizou-se para os experimentos em conjuntos de dados sintéticos de 1% a 10% de restrições para cada conjunto, e utilizou-se 10% de restrições no conjunto de dados reais por ser um conjunto razoavelmente menor, pois, nesse caso, não há como obter resultados satisfatórios com, por exemplo, 1% de restrições.

Para comprovar estatisticamente se há diferenças significantes nos resultados de desempenho dos algoritmos nos vários conjuntos de dados, foi aplicado o teste de *Friedman*, apresentado na Seção 2.1.4.2, para verificar as diferenças de desempenho considerando um certo nível de confiança. Porém, somente o teste de *Friedman* não é suficiente para dizer qual dos algoritmos obteve o melhor desempenho nos experimentos. Para isso, foi aplicado também, o teste *post-hoc Bonferroni-Dunn* (veja Seção 2.1.4.2), o qual possibilita a comparação entre pares de algoritmos, a fim de, determinar qual deles tem um desempenho superior.

A seguir são apresentados os experimentos que corroboram a afirmação de que o algoritmo criado no trabalho descrito nesta dissertação supera algoritmos semelhantes da literatura, com o objetivo de encontrar agrupamentos em conjuntos de dados de formas diversas. Antes disso, uma bateria de testes foi realizada para determinar um cenário em que o MRS-*kmeans* pudesse ter todo o seu potencial explorado. Por fim, foi realizado um estudo de caso, que também comprova a eficácia do algoritmo MRS-*kmeans* em dados reais.

5.2 Experimentos

Esta seção apresenta os resultados dos experimentos realizados com o objetivo de estressar os algoritmos testados em diferentes situações. Devido a característica de inicialização não determinística dos algoritmos testados, para cada divisão do conjunto de restrições em *folds* de treinamento e teste, como apresentado na Seção 3.3, os algoritmos de detecção de agrupamentos foram executados vinte vezes e foi capturada a média dos resultados dessas execuções. O método *cross-validation* utilizado foi o estratificado, isto significa que, para todos os conjuntos de restrições gerados foram mantidas as mesmas proporções entre as restrições *must-link* e *cannot-link* em nível de instância, mantendo a proporção também nos *folds* de treinamento e teste. Os experimentos são apresentados a seguir.

5.2.1 Variando o Parâmetro Z

No primeiro experimento o objetivo foi determinar um valor para o parâmetro Z do algoritmo MRS-*kmeans*. Como apresentado na Seção 4.3, o parâmetro Z especifica uma porcentagem de representantes auxiliares para cada grupo de um conjunto de dados, que serão utilizados durante o processo de detecção de agrupamentos. Uma quantidade relativamente grande de representantes auxiliares pode ser gerada, por se derivar das restrições entre instâncias informadas externamente ao processo, e com isso aumentar o número de cálculos de distância necessários na fase de atribuição de instâncias aos grupos. Além disso, os representantes auxiliares podem estar mal distribuídos na extensão do conjunto de dados, desta forma, pouco contribuindo para a descoberta das formas diversas da estrutura real do conjunto de dados. Para amenizar esse problema é necessário selecionar uma quantidade menor de representantes auxiliares, de forma que, esses representantes melhor contribuam para o objetivo principal do algoritmo. Tendo isso em mente, desenvolveu-se esse experimento para comparar o desempenho do algoritmo MRS-*kmeans* com os seguintes valores para Z: 10%, 20%, 30%, 40% e 50%. O experimento foi aplicado aos 8 conjuntos de dados sintéticos em 2D, utilizando 1% das instâncias do conjunto como restrições. Os resultados obtidos podem ser vistos na Figura 5.26.

Na análise desse experimento depara-se com duas situações: primeiramente, ao utilizar um baixo valor para o parâmetro Z, a acurácia do algoritmo tende a ser menor, além disso, o algoritmo demora mais para convergir, resultando em um alto custo computacional; por outro lado, ao utilizar um alto valor para o parâmetro Z, a acurácia do algoritmo aumenta e o número de iterações cai, diminuindo o custo computacional, porém, o número de cálculos de distância deixa de ser em função das iterações e passa a ser em função da quantidade de representantes auxiliares. Isto é, a medida em que o valor de Z aumenta, a tendência é que o custo computacional volte a aumentar. Dessa forma, é necessário encontrar um valor para Z que atinja um melhor "custo $\times$ benefício" para a execução do

algoritmo, procurando observar o momento em que os resultados de desempenho, tanto da acurácia quanto do custo computacional, se estabilizam.

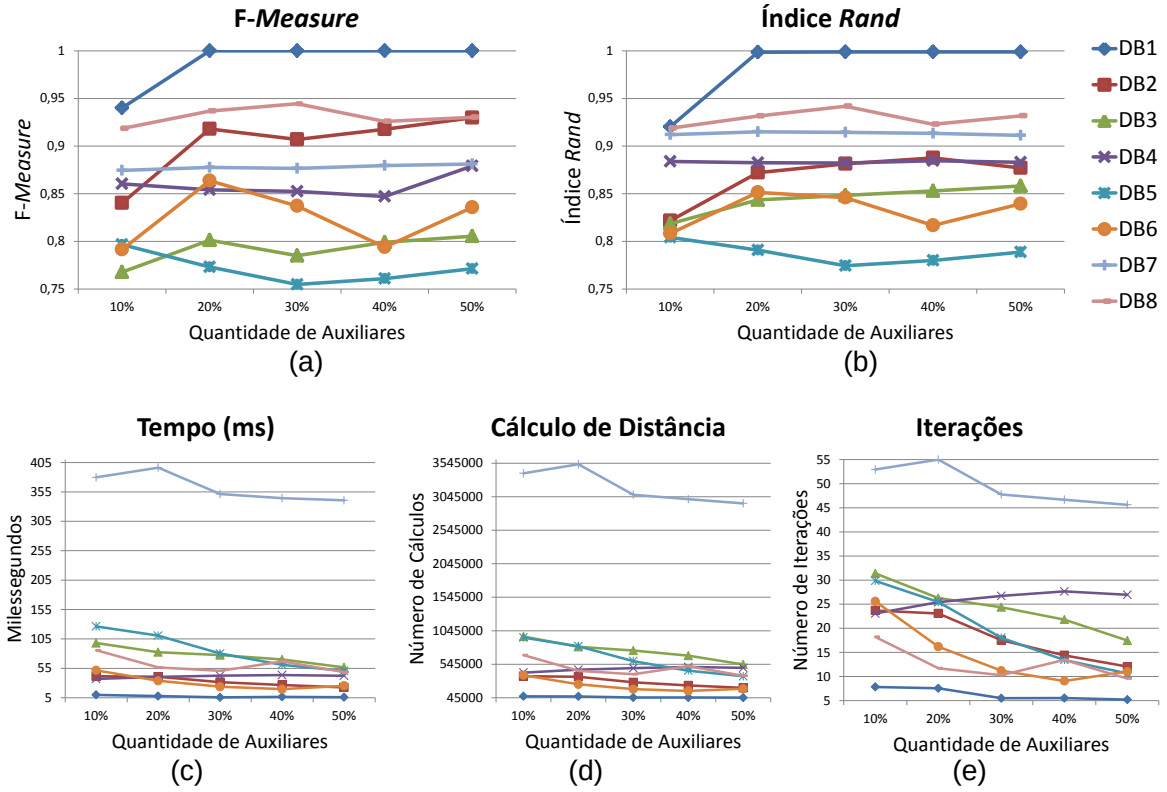

Figura 5.26 – Resultados dos experimentos que varia o valor do parâmetro Z do algoritmo MRS-*kmeans*.

Observando os resultados ilustrados na Figura 5.26, é possível notar uma tendência a estabilização na execução do MRS-*kmeans* com 30% dos representantes auxiliares. Os conjuntos DB1, DB2, DB3, DB6 e DB8 possuem um melhor custo benefício utilizando 30% dos representantes. Nos conjuntos DB4, DB5 e DB7 o aumento de representantes auxiliares não melhora sua acurácia (Figuras 5.26(a) e 5.26(b)), porém com $Z = 30\%$ o desempenho computacional na execução tente a ser melhor (Figuras 5.26(c), 5.26(d) e 5.26(e)), como exemplo, pode-se considerar o custo computacional do algoritmo no conjunto DB7. Esse custo computacional, alto para uma pequena quantidade de representantes auxiliares, está relacionado ao número de iterações necessárias para o algoritmo convergir, isto é, quanto maior a quantidade de iterações necessárias, maior número de cálculos de distância será realizado. Pode-se concluir desta forma que manter o parâmetro Z em 30% proporciona um melhor custo $\times$ benefício para a execução do algoritmo.

5.2.2 Variando a Porcentagem de Restrições

O objetivo do segundo experimento foi encontrar uma quantidade de restrições razoavelmente pequena na qual o algoritmo MRS-*kmeans* mostrasse um bom desempenho. Em

vários trabalhos da literatura de semi-supervisão, autores definem a quantidade de restrições em 10%, 20% ou até 30% das instâncias do conjunto de dados. Porém, dependendo do tamanho do conjunto, uma quantidade muito grande de informação adicional é fornecida, e isso não é um cenário ideal para algoritmos de semi-supervisão. Com isso, deseja-se mostrar com esse experimento que o algoritmo MRS-*kmeans* tem um bom desempenho mesmo em pequenas quantidades de informação adicional, mostrando ser superior a outros algoritmos com apenas 1% do conjunto de dados em forma de restrições. Para realizar esse experimento variou-se a porcentagem de restrições entre 1% e 10% das instâncias do conjunto de dados, fixando Z em 30%. O resultado, ilustrado na Figura 5.27, mostra o desempenho sob o índice *Rand*, o qual mostra a taxa de acerto da real estrutura dos dados, e a Figura 5.28 mostra o tempo utilizado para execução dos diferentes experimentos. Os resultados são discutidos a seguir.

De maneira geral, os resultados ilustrados na Figura 5.27 mostram que na maioria dos casos o algoritmo MRS-*kmeans* tem um desempenho bastante superior ao COP-*kmeans* e ao MLC-*kmeans*, mesmo com 1% de restrições. Também é possível notar que o aumento dessa porcentagem não afeta seu desempenho nesse quesito. Enquanto isso, tanto o COP-*kmeans* como o MLC-*kmeans* têm uma leve tendência a aumentar o seu desempenho conforme aumenta-se as restrições, mas não chega a alcançar o desempenho do MRS-*kmeans*. Nos conjuntos DB2, DB4, DB5 e DB6 houve uma variação de desempenho aos diferentes números de restrições, porém, nos conjuntos DB1, DB3, DB7 e DB8 os resultados se mantiveram mais estáveis. Contudo, pode-se perceber que o resultado do algoritmo MRS-*kmeans* é bom mesmo utilizando uma quantidade considerável de restrições, por exemplo 10% de restrições. Entretanto, o seu custo computacional piora a medida que mais restrições são consideradas, como mostra a Figura 5.28.

Os resultados ilustrados na Figura 5.28 mostram o tempo de execução, em milissegundos, dos algoritmos nas diferentes combinações de conjuntos de dados e conjuntos de restrições. Nesse caso, o tempo de execução leva em consideração o número de cálculos de distância, a quantidade de iterações necessárias até o algoritmo convergir e o número de falhas. De modo geral, pode-se perceber que o algoritmo MRS-*kmeans* tem um tempo bem inferior aos demais algoritmos, considerando 1% de restrições, e esse tempo vai aumentando conforme aumentam as restrições, de forma que, em alguns casos chega a ultrapassar o tempo gasto por outros algoritmos, como no DB1, DB6, DB7 e DB8, considerando 10% de restrições. Entretanto, devido as inúmeras falhas, que aumentam conforme o aumento das restrições nos algoritmo COP-*kmeans* e MLC-*kmeans*, o processo acaba sendo mais demorado que o MRS-*kmeans*, e como visto na Figura 5.27, também acaba sendo menos eficiente.

A partir desses experimentos é possível definir um cenário em que o algoritmo MRS-*kmeans* se destaca. Definindo o parâmetro Z em 30% e com apenas 1% das instâncias do conjunto de dados contribuindo para restrições é possível ter um resultado bastante

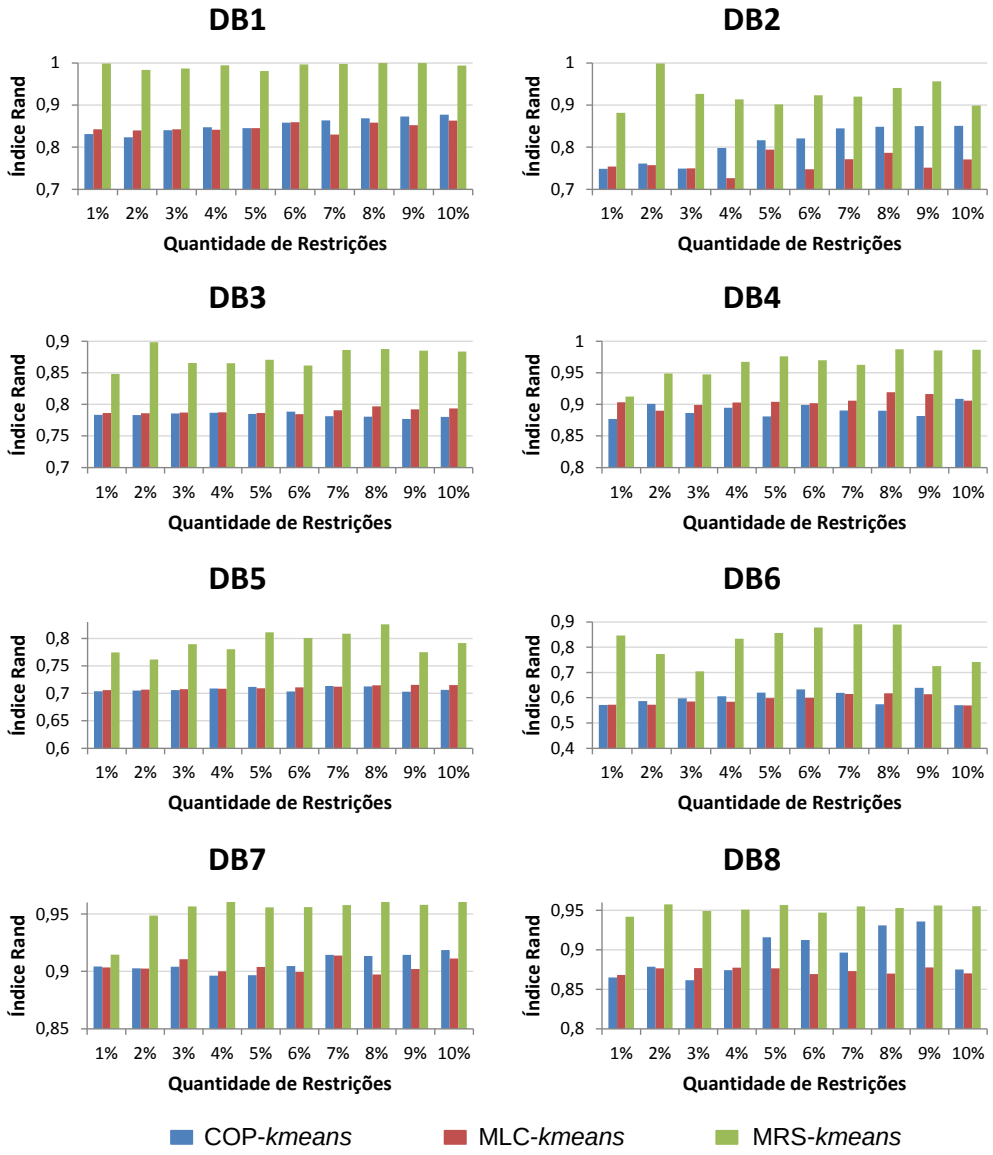

Figura 5.27 – Resultados do índice *Rand* no experimento que varia a quantidade de restrições nos algoritmos COP-*kmeans*, MLC-*kmeans* e MRS-*kmeans*.

superior aos algoritmos da literatura. A próxima seção apresenta uma comparação dessa configuração nas diferentes dimensionalidades dos conjuntos de dados sintéticos.

5.2.3 Variando a Dimensionalidade

Nesse terceiro experimento com os conjuntos de dados sintéticos o objetivo foi mostrar o desempenho dos algoritmos nos conjuntos de dados variando suas dimensionalidades. Para isso, foi utilizada a ferramenta *Distribution Painter* para manter as estruturas originais dos conjuntos de duas dimensões para os conjuntos com 3, 4, 6 e 8 dimensões. Lembrando que a quantidade de instâncias também aumenta conforme o número de dimensões desses novos conjuntos, como apresentado na Tabela 5.2.

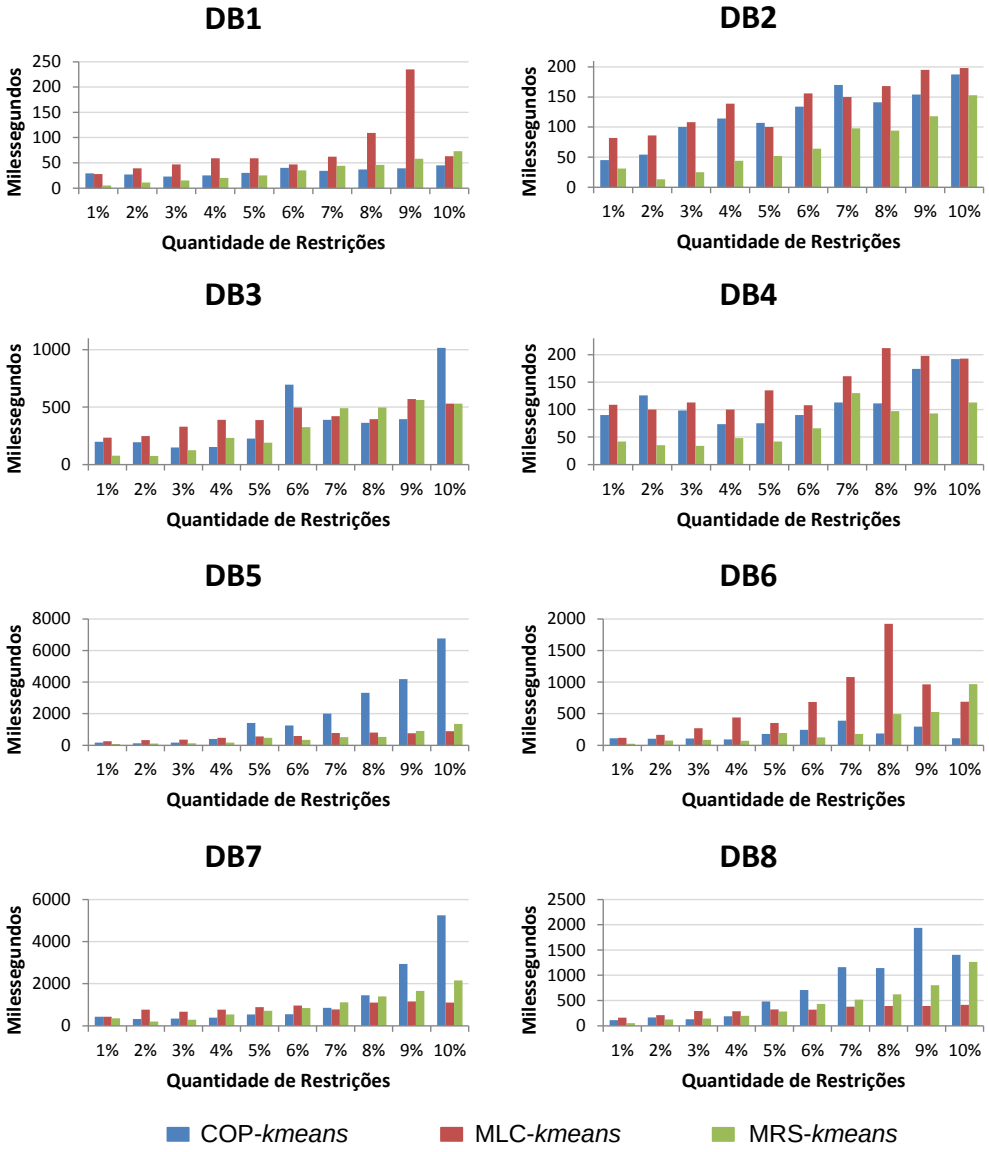

Figura 5.28 – Resultados de tempo de execução no experimento que varia a quantidade de restrições nos algoritmos COP-*kmeans*, MLC-*kmeans* e MRS-*kmeans*.

Por questões de organização, foram divididas nas Figuras 5.29 e 5.30 os resultados dos experimentos considerando os quarenta conjuntos de dados. Na Figura 5.29 pode-se perceber que o algoritmo MRS-*kmeans* continua tendo um melhor resultado nas diferentes configurações de cada conjunto de dados. É necessário enfatizar que, apesar de tentar manter a estrutura original dos conjuntos de duas dimensões na geração de conjuntos de maior dimensionalidade, na verdade estão sendo gerados conjuntos muito diferentes. Por esse motivo, não é possível compará-los entre si, isto é, cada conjunto, dependendo da sua dimensionalidade, é um conjunto diferente mesmo mantendo a mesma estrutura de agrupamento.

Analisando os índices F-*Measure* e *Rand*, nos resultados dos conjuntos DB1 e DB2, é possível ver que o algoritmo MRS-*kmeans* conseguiu ser superior considerando 2, 3,

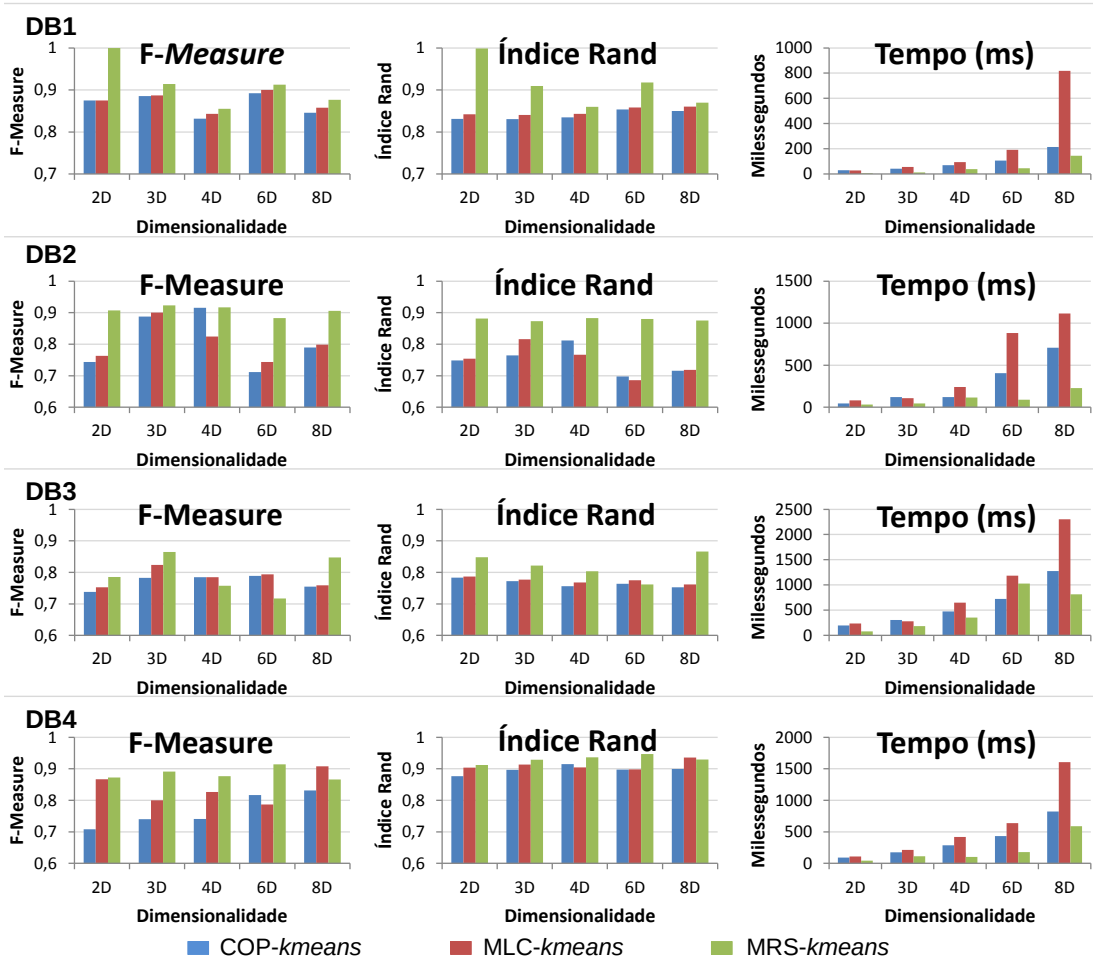

Figura 5.29 – Resultados dos experimentos variando a dimensionalidade dos conjuntos DB1, DB2, DB3 e DB4.

4, 6 e 8 dimensões. Por exemplo, para o conjunto de dados DB1 de duas dimensões, considerando a métrica F-*Measure*, o MRS-*kmeans* apresentou um desempenho 12,5% superior para ambos os algoritmos COP-*kmeans* e MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 81,02% menos tempo do que o COP-*kmeans* e 80,28% menos tempo que o MLC-*kmeans*. Para o conjunto de dados DB2 de seis dimensões o MRS-*kmeans* apresentou um desempenho 20,7% superior, considerando o índice *Rand*, quando comparado com o COP-*kmeans* e 22,06% superior quando comparado com o MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 77,98% menos tempo do que o COP-*kmeans* e 89,88% menos tempo que o MLC-*kmeans*.

Nos conjuntos DB3 e DB4, o MRS-*kmeans* foi superado em algumas configurações dos conjuntos, mas teve um desempenho superior na maioria deles. Por exemplo, para o conjunto de dados DB3 de oito dimensões o MRS-*kmeans* apresentou um desempenho 13,06% superior, considerando o índice *Rand*, quando comparado com o COP-*kmeans* e 12,02% superior quando comparado com o MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 36,13% menos tempo do que o COP-*kmeans* e 64,58% menos

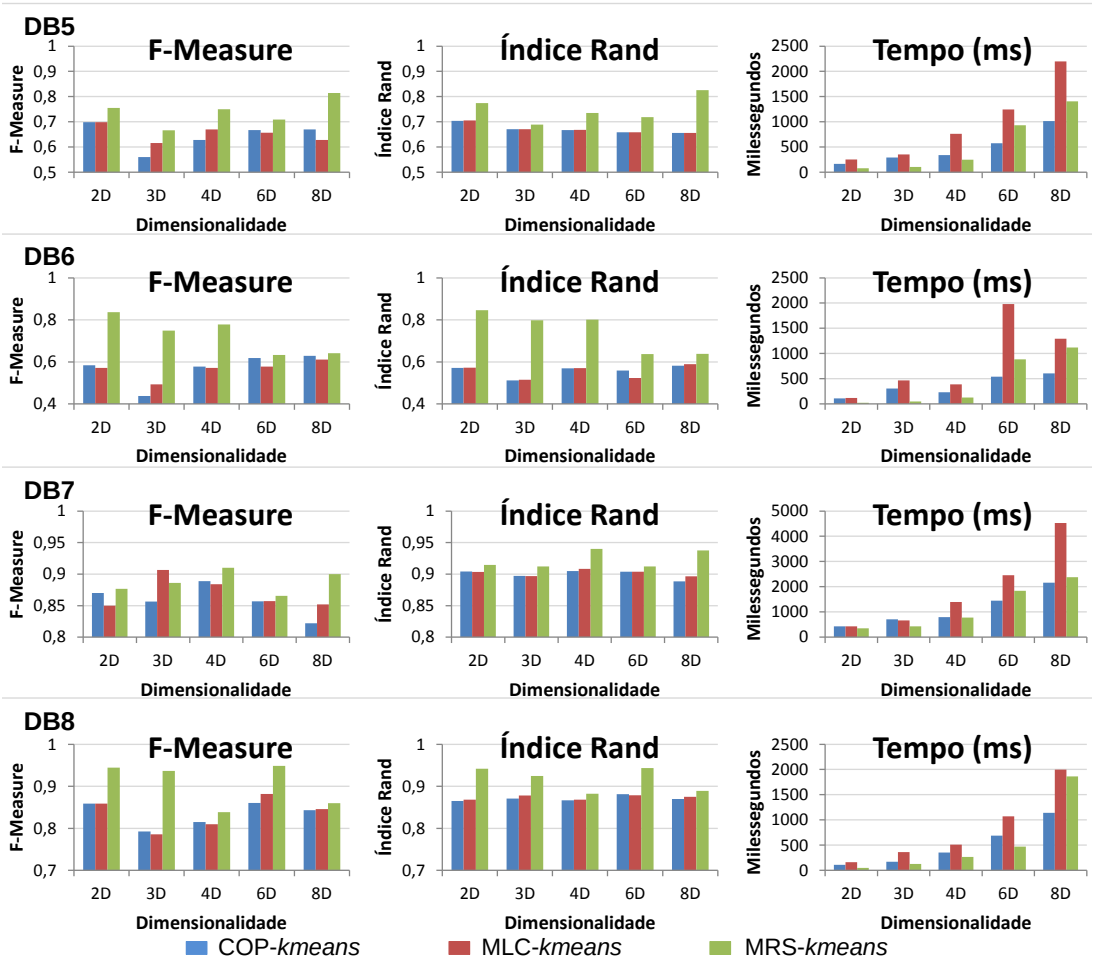

Figura 5.30 – Resultados dos experimentos variando a dimensionalidade dos conjuntos DB5, DB6, DB7 e DB8.

tempo que o MLC-*kmeans*. Já para o conjunto de dados DB4 de 3 dimensões o MRS-*kmeans* apresentou um desempenho 16,94% superior, considerando a métrica F-*Measure*, quando comparado com o COP-*kmeans* e 10,21% superior quando comparado com o MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 36,93% menos tempo do que o COP-*kmeans* e 48,13% menos tempo que o MLC-*kmeans*.

Na Figura 5.30 são apresentados os resultados para os outros quatro conjuntos, sendo que o algoritmo MRS-*kmeans* conseguiu bons resultados nos conjuntos DB5, DB6 e DB8 para todas as configurações desses conjuntos de dados. Para o conjunto DB7, o resultado foi superior na grande maioria das configurações, sendo superado em alguns poucos conjuntos. Por exemplo, para o conjunto de dados DB5 de 8 dimensões o MRS-*kmeans* apresentou um desempenho 17,68% superior, considerando a métrica F-*Measure*, quando comparado com o COP-*kmeans* e 22,82% superior quando comparado com o MLC-*kmeans*. Porém, o MRS-*kmeans* demandou 27,93% mais tempo do que o COP-*kmeans*, mas em relação ao tempo do MLC-*kmeans*, demandou 52,32% menos tempo. No conjunto de dados DB6 de 2 dimensões o MRS-*kmeans* apresentou um desempenho 32,49% supe-

rior, considerando o índice *Rand*, quando comparado com o COP-*kmeans* e 32,36% superior quando comparado com o MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 78,25% menos tempo do que o COP-*kmeans* e 79,99% menos tempo que o MLC-*kmeans*. Para o conjunto de dados DB7 de 8 dimensões o MRS-*kmeans* apresentou um desempenho 8,67% superior, considerando a métrica F-*Measure*, quando comparado com o COP-*kmeans* e 5,35% superior quando comparado com o MLC-*kmeans*. Porém, o MRS-*kmeans* demandou 9,2% mais tempo do que o COP-*kmeans*, mas em relação ao tempo do MLC-*kmeans*, demandou 47,42% menos tempo. Já para o conjunto de dados DB8 de 3 dimensões o MRS-*kmeans* apresentou um desempenho 15,37% superior, considerando a métrica F-*Measure*, quando comparado com o COP-*kmeans* e 16,13% superior quando comparado com o MLC-*kmeans*. Com relação ao tempo, o MRS-*kmeans* demandou 27,16% menos tempo do que o COP-*kmeans* e 65,19% menos tempo que o MLC-*kmeans*.

Nesta Figura 5.30 pode-se perceber que conforme aumenta a dimensionalidade dos conjuntos e consequentemente a quantidade de instâncias, o MRS-*kmeans* tem um aumento considerável no tempo de execução em comparação aos experimentos nos conjuntos de duas dimensões. Porém, devido ao menor número de iterações, o tempo de execução, de maneira geral ainda é melhor quando comparado com os algoritmos COP-*kmeans* e MLC-*kmeans*.

Pode-se concluir com esse experimento que o algoritmo MRS-*kmeans* consegue manter um desempenho satisfatório mesmo aumentando a quantidade de dimensões e de instâncias em comparação aos algoritmos COP-*kmeans* e MLC-*kmeans*. Entretanto, ainda busca-se estudar o seu comportamento em conjuntos de dados de alta dimensionalidade, identificando cenários nos quais o MRS-*kmeans* manteria o seu desempenho superior aos demais algoritmos que adotam a abordagem de particionamento de dados e com isso, apontando possíveis melhorias. A próxima seção apresenta uma comparação mais detalhada entre os três algoritmos testados, além disso, comprovando suas diferenças de desempenho por meio de avaliações estatísticas.

5.2.4 Comparação entre os Algoritmos

Esta seção apresenta os resultados dos experimentos que comprovam tanto a eficácia quanto a eficiência do algoritmo MRS-*kmeans* em comparação com os algoritmos COP-*kmeans* e MLC-*kmeans*. Nesse contexto, entenda eficácia como o quanto o algoritmo é capaz de encontrar a verdadeira estrutura de um conjunto de dados, e também, entenda eficiência como o quanto o algoritmo minimiza o custo computacional necessário para adquirir tal resultado. Desse modo, a avaliação da eficácia dos algoritmos foi abordada de duas formas: primeiramente, foi avaliado o quanto os agrupamentos se aproximaram da real estrutura dos dados, por meio do índice *Rand*; logo em seguida, foi avaliado o quão satisfeitas foram as restrições, por meio da F-*Measure*. Nesses experimentos foi fixado o

parâmetro Z em 30% e utilizados conjuntos de restrições com 1% do tamanho de cada um dos oito conjuntos de dados em duas dimensões. Os resultados são discutidos a seguir.

Em relação a eficaz descoberta da real estrutura dos dados, a Figura 5.31 apresenta os resultados de desempenho dos algoritmos nos diferentes conjuntos de dados, em duas dimensões, pelo índice *Rand*. Nesse experimento pode-se notar que o algoritmo MRS-*kmeans* teve melhor desempenho que os demais algoritmos em todos os conjuntos testados, destacando-se nos resultados dos conjuntos DB1, DB2 e DB6, com mais de 15% de acerto em relação aos demais algoritmos. Para esse experimento, a Figura 5.34 ilustra de melhor forma o resultado do MRS-*kmeans* em comparação aos outros algoritmos. Nessa figura cada cor representa um grupo do particionamento gerado por cada algoritmo, sendo que, o ideal é que cada forma seja preenchida com apenas uma cor. É importante destacar também que para o experimento da Figura 5.34 foram utilizados os mesmos centróides iniciais, portanto, todos os algoritmos são inicializados da mesma forma. Os resultados do MRS-*kmeans* nos conjuntos DB1, DB2, DB3 e DB8 ficaram muito próximos a estrutura real dos dados, nos demais conjuntos DB4, DB5, DB6 e DB7, o resultado não foi tão expressivo, mas em comparação aos resultados dos outros algoritmos o desempenho foi bastante superior.

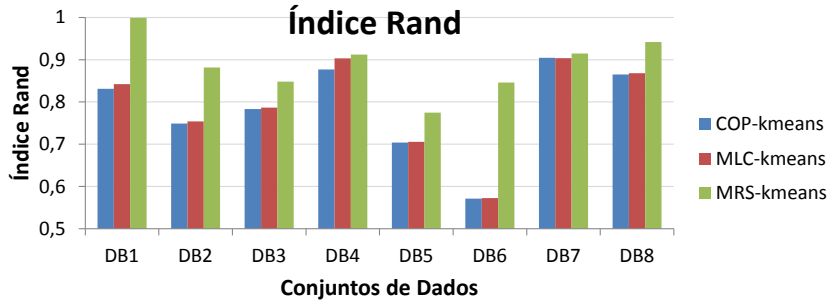

Figura 5.31 – Desempenho do algoritmo MRS-*kmeans* em comparação com os algoritmos COP-*kmeans* e MLC-*kmeans* por meio do índice *Rand*.

Em relação a eficácia em satisfazer as restrições, a Figura 5.32 apresenta os resultados dos algoritmos na metodologia de avaliação específica para algoritmos de semi-supervisão (veja Seção 3.3) por meio da métrica F-*Measure*. Os resultados foram bastante parecidos com os apresentados na Figura 5.31, novamente o MRS-*kmeans* foi melhor em todos os conjuntos testados, se destacando nos conjuntos DB1, DB2 e DB6. É possível concluir com esse experimento que o algoritmo MRS-*kmeans* satisfaz mais restrições que os outros algoritmos.

Já a eficiência com relação ao custo computacional dos algoritmos é ilustrada nas Figuras 5.33(a), 5.33(b) e 5.33(c), onde são considerados os principais fatores para o custo de execução dos algoritmos. Com exceção do conjunto de dados DB7, o algoritmo MRS-*kmeans* conseguiu retornar o agrupamento com menor número de cálculos de distância em todos os conjunto de dados, como mostra a Figura 5.33(a). Da mesma forma ocorre para

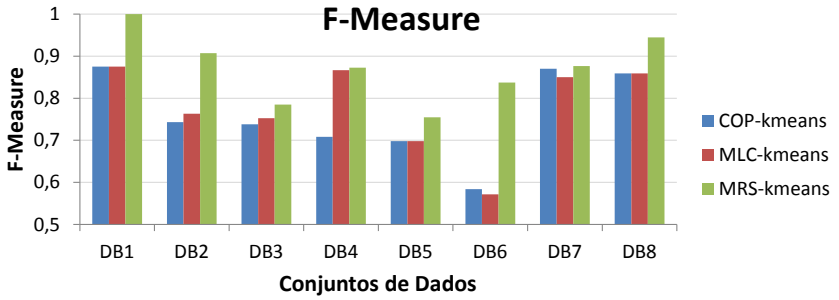

Figura 5.32 – Desempenho do algoritmo MRS-*kmeans* em comparação com os algoritmos COP-*kmeans* e MLC-*kmeans* por meio da F-*Measure*.

o número de iterações executadas até o algoritmo convergir, o MRS-*kmeans* retornou com menor número de iterações na maioria dos conjuntos testados, como ilustrado na Figura 5.33(b). Apesar do algoritmo MRS-*kmeans* realizar naturalmente mais cálculos de distância que os demais algoritmos, por causa dos vários representantes adicionais, a sua rápida convergência supera essa desvantagem, já que ao realizar menos iterações são necessários menores números de cálculos de distância, o que reflete no tempo total de execução dos algoritmos, ilustrado na Figura 5.33(c). Outro fator a se considerar, ao observar os resultados, está relacionado a quantidade de falhas geradas pelos algoritmos COP-*kmeans* e MLC-*kmeans*, causada por uma característica discutida na Seção 3.2.1. Essas falhas também influenciam negativamente no tempo total de execução para os dois algoritmos em relação ao MRS-*kmeans*, o qual não possui essa deficiência. Enfim, o tempo de execução mostra que o MRS-*kmeans* não só é eficaz como também realiza seu trabalho consumindo menos recursos.

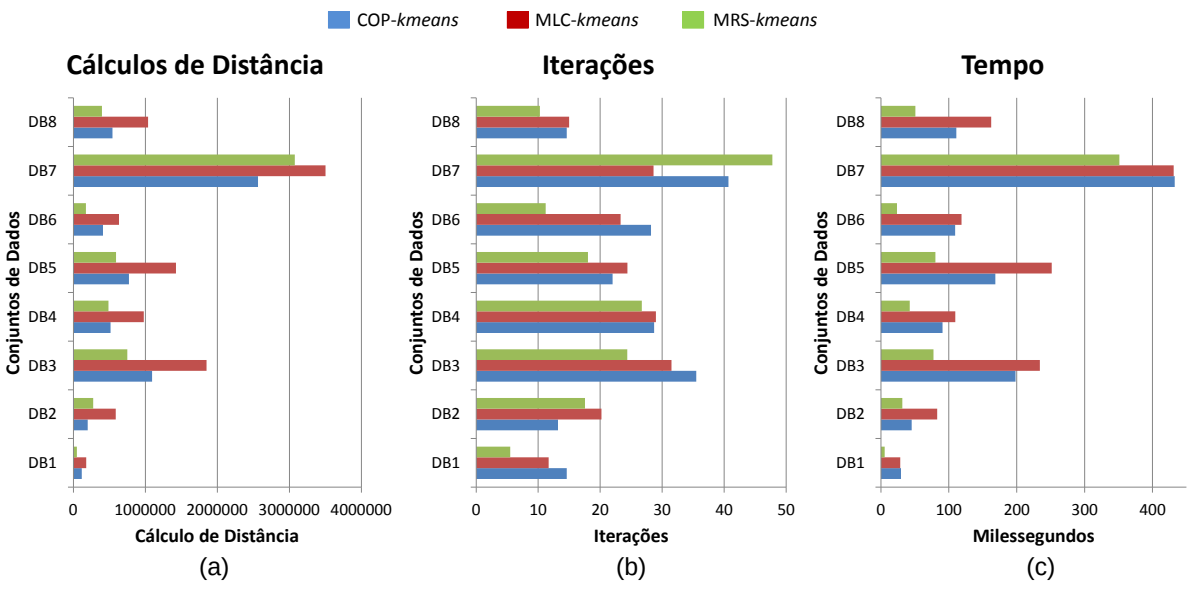

Figura 5.33 – Custo computacional dos experimentos realizados.

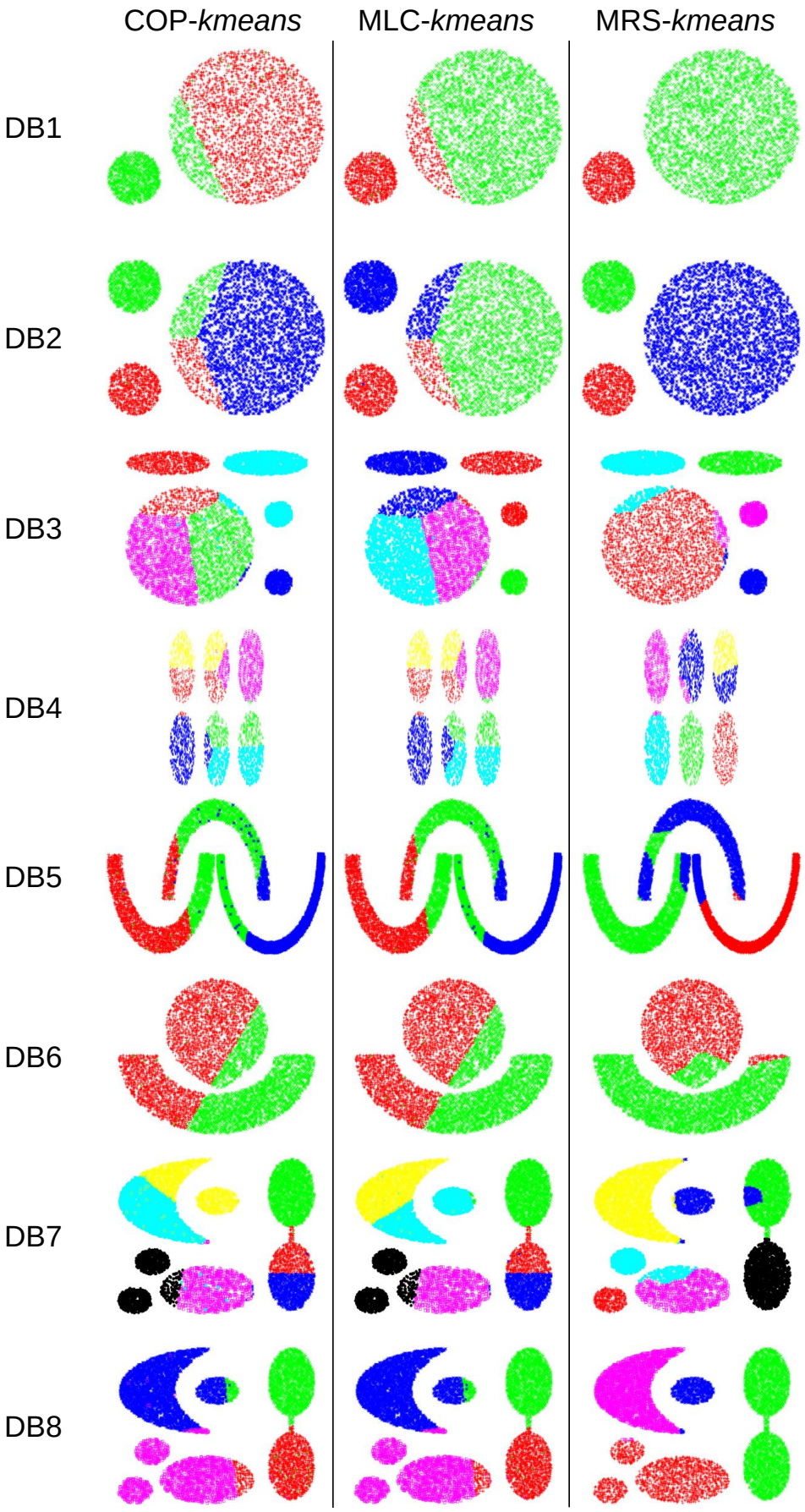

Figura 5.34 – Resultado do particionamento, em que cada cor representa um grupo diferente.

Teste Estatístico

Para realizar o teste estatístico de comparação entre os algoritmos foram utilizados os resultados dos experimentos considerando os quarenta conjuntos de dados sintéticos, utilizando a metodologia específica para semi-supervisão com a métrica F-*Measure*. Os resultados desse experimento estão descritos na Tabela 5.3, a qual ranqueia os resultados de desempenho dos algoritmos a fim de compor o teste de *Friedman*, que determina se existe uma diferença significativa de desempenho entre os algoritmos testados. A Tabela 5.3 também traz o valor da soma dos *ranks* e a média dos *ranks* do desempenho de cada algoritmo, que são usados durante o teste estatístico de *Friedman* e *post-hoc Bonferroni-Dunn* (mais detalhes na Seção 2.1.4.2).

Nesse experimento, busca-se responder a seguinte questão: com $\alpha = 0,05$, ou seja, com 95% de confiança pode-se concluir que o algoritmo MRS-*kmeans* supera os algoritmos COP-*kmeans* e MLC-*kmeans*? Para isso, formula-se uma hipótese nula H_0 que diz: "Os três algoritmos têm o mesmo desempenho". Formula-se também uma hipótese alternativa H_A, a qual poderá ser avaliada se H_0 for rejeitada, que diz: "O algoritmo MRS-*kmeans* tem desempenho superior aos outros algoritmos". A fim de testar se a hipótese nula H_0 é rejeitada aplicou-se o teste de *Friedman*, como apresentado a seguir.

O primeiro passo é calcular o valor estatístico de *Friedman* χ_r^2, o qual considera o número de conjuntos de dados N, o número de algoritmos testados t e os valores da soma dos *ranks* de cada algoritmo (veja Tabela 5.3). O cálculo desse valor estatístico pode ser visto logo abaixo.

$$\chi_r^2 = \frac{12}{Nt\,(t+1)}\left[\sum_{j=1}^{t} P_j^2 - 3N\,(t+1)\right] = \frac{12}{40\cdot 3\,(3+1)}\left(104^2+90^2+46^2\right) - 3\cdot 40\,(3+1)$$

$$\chi_r^2 = 0,025\,(21.032) - 480 = 525,8 - 480 = \mathbf{45,8}$$

Os autores em (IMAN; DAVENPORT, 1980) mostraram que o teste de *Friedman* é indesejavelmente conservador e necessitaria de uma melhor estatística. Por esse motivo, a estatística de *Friedman* será usada para compor o seguinte cálculo.

$$F_F = \frac{(N-1)\,\chi_r^2}{N\,(t-1) - \chi_r^2} = \frac{(40-1)\,45,8}{40\,(3-1) - 45,8} = \frac{1.786,2}{80 - 45,8} = \frac{1.786,2}{34,2} = \mathbf{52,228}$$

Esse teste é aplicado considerando $t-1$ e $(t-1)\,(N-1)$ graus de liberdade. Seu valor crítico, obtido em tabela de valores críticos da F-*Distribution* é 3,1137. Parte da tabela de valores críticos por ser vista em (ZAR, 2007) e no Apêndice A.1. Esse resultado afirma que existe uma diferença significativa entre o desempenho dos algoritmos, pois $3,1137 < 52,228$, e neste caso, rejeita-se a hipótese nula. Entretanto, o teste de *Friedman* só pode definir se existe uma diferença significativa entre algoritmos, mas não indica qual dos algoritmos teve o desempenho superior. Para isso, usa-se um teste *post-hoc*

Tabela 5.3 – *Ranks* de desempenho da F-*Measure* para o teste de *Friedman*.

Conjuntos de Dados		COP-*kmeans*	MLC-*kmeans*	MRS-*kmeans*
2D	DB1	0,8750 (2,5)	0,8750 (2,5)	1,0000 (1)
	DB2	0,7433 (3)	0,7633 (2)	0,9070 (1)
	DB3	0,7381 (3)	0,7524 (2)	0,7851 (1)
	DB4	0,7083 (3)	0,8667 (2)	0,8725 (1)
	DB5	0,6982 (2,5)	0,6982 (2,5)	0,7548 (1)
	DB6	0,5839 (2)	0,5714 (3)	0,8371 (1)
	DB7	0,8700 (2)	0,8500 (3)	0,8766 (1)
	DB8	0,8589 (2,5)	0,8589 (2,5)	0,9445 (1)
3D	DB1	0,8857 (3)	0,8868 (2)	0,9143 (1)
	DB2	0,8875 (3)	0,9000 (2)	0,9231 (1)
	DB3	0,7823 (3)	0,8234 (2)	0,8646 (1)
	DB4	0,7400 (3)	0,8000 (2)	0,8910 (1)
	DB5	0,5604 (3)	0,6163 (2)	0,6668 (1)
	DB6	0,4382 (3)	0,4936 (2)	0,7493 (1)
	DB7	0,8566 (3)	0,9066 (1)	0,8859 (2)
	DB8	0,7926 (2)	0,7855 (3)	0,9366 (1)
4D	DB1	0,8319 (3)	0,8431 (2)	0,8550 (1)
	DB2	0,9155 (2)	0,8245 (3)	0,9168 (1)
	DB3	0,7844 (1,5)	0,7844 (1,5)	0,7573 (3)
	DB4	0,7405 (3)	0,8262 (2)	0,8764 (1)
	DB5	0,6280 (3)	0,6695 (2)	0,7499 (1)
	DB6	0,5781 (2)	0,5710 (3)	0,7780 (1)
	DB7	0,8889 (2)	0,8839 (3)	0,9102 (1)
	DB8	0,8149 (2)	0,8096 (3)	0,8387 (1)
6D	DB1	0,8923 (3)	0,9000 (2)	0,9127 (1)
	DB2	0,7113 (3)	0,7433 (2)	0,8827 (1)
	DB3	0,7886 (2)	0,7939 (1)	0,7171 (3)
	DB4	0,8167 (2)	0,7867 (3)	0,9138 (1)
	DB5	0,6677 (2)	0,6569 (3)	0,7091 (1)
	DB6	0,6182 (2)	0,5773 (3)	0,6332 (1)
	DB7	0,8569 (3)	0,8570 (2)	0,8654 (1)
	DB8	0,8604 (3)	0,8819 (2)	0,9485 (1)
8D	DB1	0,8460 (3)	0,8577 (2)	0,8768 (1)
	DB2	0,7893 (3)	0,7988 (2)	0,9055 (1)
	DB3	0,7548 (3)	0,7588 (2)	0,8474 (1)
	DB4	0,8312 (3)	0,9081 (1)	0,8660 (2)
	DB5	0,6703 (2)	0,6284 (3)	0,8143 (1)
	DB6	0,6291 (2)	0,6109 (3)	0,6417 (1)
	DB7	0,8219 (3)	0,8518 (2)	0,9000 (1)
	DB8	0,8430 (3)	0,8457 (2)	0,8601 (1)
Soma dos *Ranks*		104	90	46
Média dos *Ranks*		2,6	2,25	1,15

que complementa o resultado do teste de *Friedman* na avaliação de desempenho dos algoritmos.

Para este caso em especifico, optou-se por usar o teste *post-hoc Bonferroni-Dunn*, pois com ele é possível fixar em um único algoritmo de controle e compará-lo aos demais. O primeiro passo ao realizar esse teste é definir a diferença crítica (CD) de desempenho dos algoritmos, delimitando um valor limite para dizer se um algoritmo é superior a outro ou não. O cálculo da diferença crítica é dado a seguir.

$$CD = q_\alpha \sqrt{\frac{t\,(t+1)}{6N}} = 2,241 \sqrt{\frac{3 \cdot 4}{6 \cdot 40}} = 2,241 \sqrt{0,05} = 2,241 \cdot 0,2236 = \mathbf{0,501}$$

O valor crítico q_α é dado baseado na *Studentized Range Statistic*, dividida por $\sqrt{2}$. Essa tabela pode ser encontrada em (DEMSAR, 2006) e no Apêndice A.2. Com o valor CD calculado basta apenas calcular a diferença do *rank* médio entre cada par de algoritmos. A primeira comparação é feita entre o COP-*kmeans* e o MRS-*kmeans*, o nosso algoritmo de controle.

$$z_{COP \times MRS} = \frac{\left(\bar{P}_{COP} - \bar{P}_{MRS}\right)}{\sqrt{\frac{t(t+1)}{6N}}} = \frac{2,6 - 1,15}{\sqrt{\frac{3(3+1)}{6 \cdot 40}}} = \frac{1,45}{0,05} = \mathbf{29}$$

Como o valor calculado é maior que a diferença crítica (0,501 < 29), pode-se concluir que o algoritmo MRS-*kmeans* é estatisticamente superior ao algoritmo COP-*kmeans*. A segunda comparação é feita entre os algoritmos MLC-*kmeans* e MRS-*kmeans*, como mostrado a seguir.

$$z_{MLC \times MRS} = \frac{\left(\bar{P}_{MLC} - \bar{P}_{MRS}\right)}{\sqrt{\frac{t(t+1)}{6N}}} = \frac{2,25 - 1,15}{\sqrt{\frac{3(3+1)}{6 \cdot 40}}} = \frac{1,1}{0,05} = \mathbf{22}$$

Como o valor calculado é maior que a diferença crítica (0,501 < 22), pode-se concluir que o algoritmo MRS-*kmeans* é também estatisticamente superior ao algoritmo MLC-*kmeans*. Baseado nesses testes estatísticos é possível afirmar de fato que o desempenho do algoritmo MRS-*kmeans* é superior, considerando a qualidade dos agrupamentos gerados, para esses conjuntos de dados sintéticos com estruturas de formas diversas.

5.2.5 Estudo de Caso

Atualmente, o Brasil é um país que, devido a sua grandeza em termos geográficos, possui diferentes divisões territoriais em diferentes níveis, sendo os principais: macrorregiões, estados e microrregiões. Contudo, algumas divisões, por exemplo em microrregiões, podem não representar um conjunto de cidades de semelhantes características sociais e econômicas, o que pode gerar certa dificuldade na aplicação de políticas públicas em contextos mais específicos como são os das microrregiões. Além disso, um novo reagrupamento dessas cidades em novas microrregiões poderia ocasionar um problema que está relacionado ao agrupamento de cidades em diferentes estados, como ocorre no estudo feito em (CARVALHO et al., 2008). A Constituição Federal prevê que organizações administrativas (como, regiões metropolitanas, microrregiões, entre outras) possam ser definidas dentro de cada Estado para permitir a coordenação regionalizada da gestão de funções públicas municipais. Portanto, o objetivo do estudo de caso consistiu em avaliar o desempenho do MRS-*kmeans* como uma ferramenta que possa contribuir na solução do problema

da integração regional de municípios brasileiros em microrregiões, restringindo os limites territoriais de cada estado, para facilitar o planejamento e a aplicação de políticas públicas de âmbito espacial/regional. Nesse contexto, seria interessante levar em consideração não apenas a integração de municípios vizinhos, mas também a homogeneidade dos municípios quanto a outros fatores, por exemplo, sócio-econômicos.

Para avaliar a contribuição do MRS-*kmeans* para a solução desse problema, foram utilizados dados dos 5.596 municípios do Brasil, dividindo cada experimento para as cinco macrorregiões. Os conjuntos de dados[1] para esses experimentos foram extraídos da base de dados do Instituto de Pesquisa Econômica Aplicada (IPEA, 2015) do ano 2000, compostos por cinco tipos de atributos sobre as cidades brasileiras: os dois primeiros atributos são referentes a latitude e a longitude de cada município, para tentar simular o agrupamento por vizinhança e manter uma certa contiguidade ao agrupamento dos municípios; e outros três atributos referentes aos dados do IDHM-Longevidade, IDHM-Educação, IDHM-Renda, que compõem os dados sócio-econômicos dos municípios. Nos dois primeiros atributos desse conjunto de dados, latitude e longitude, foi aplicada uma normalização aos dados, com o intuito de evitar que o valor de um atributo predomine sobre outro ao realizar o processo de agrupamento. Essa normalização alterou os valores dos dois primeiros atributos, calculando-os na faixa entre 0 (zero) e 1 (um), como mostra a Equação 5.23, garantindo que a respectiva proporção será mantida. Na Equação 5.23, $maiorValor$ e $menorValor$ são, respectivamente, os maiores e menores valores encontrados em todo conjunto de dados para cada dimensão(FACELI et al., 2011).

$$x_{novo} = \frac{x_{atual} - menorValor}{maiorValor - menorValor} \tag{5.23}$$

Para esses experimentos, cada um dos algoritmos (COP-*kmeans*, MLC-*kmeans* e MRS-*kmeans*) foi executado 50 vezes por causa das características de inicialização aleatória. Ao final, foram obtidas as médias de desempenho dessas execuções pelo índice *Rand*. Os conjuntos de restrições foram gerados em 10% das instâncias de cada conjunto de dados, de forma que municípios do mesmo estado formavam uma restrição *must-link*, e de estados diferentes formavam uma restrição *cannot-link*. Os experimentos consistiram em agrupar as cidades nas quantidades de estados de cada macrorregião. A figura 5.35 apresenta os resultados desses experimentos.

Os resultados mostram que o algoritmo MRS-*kmeans* tem acurácia equivalente ou superior na divisão dos estados por macrorregiões, considerando principalmente informações sócio-econômicas de municípios, em relação aos algoritmos COP-*kmeans* e MLC-*kmeans*. É importante notar que as regiões sul e sudeste são mais difíceis de particionar geograficamente de forma correta devido ao pareamento sócio-econômico de seus municípios, os quais tem os maiores índices de IDHM do país. A Figura 5.36 mostra um exemplo do resultado de uma execução para essa macrorregião. Para esses experimentos, no conjunto

[1] Conjuntos de dados disponível em https://goo.gl/KrdFs0

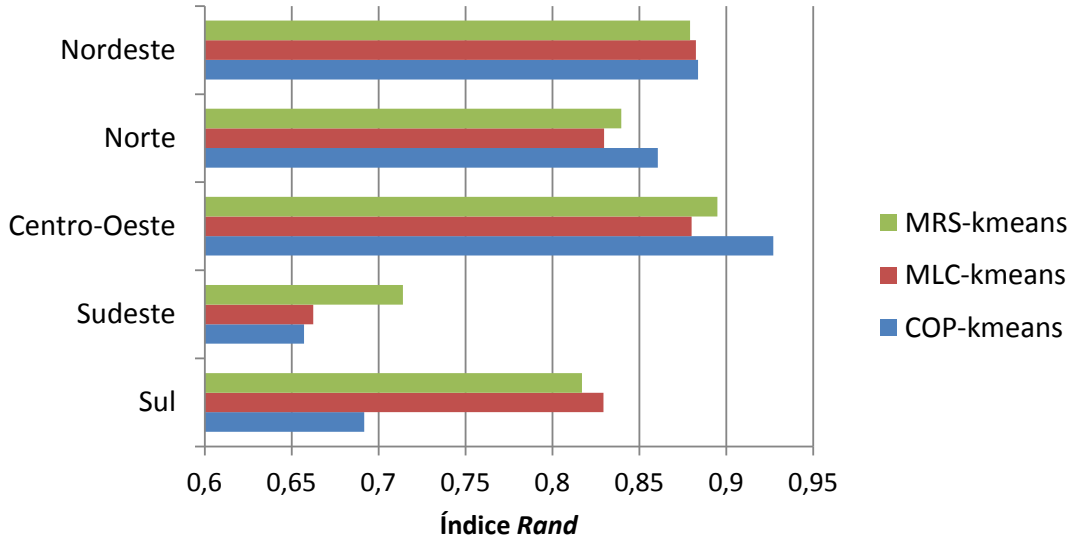

Figura 5.35 – Resultados dos particionamentos dos estados por macro regiões.

de dados "Sudeste" o MRS-*kmeans* apresentou um desempenho 7,96% superior quando comparado com o COP-*kmeans* e 7,22% superior quando comparado com o MLC-*kmeans*. Já para o conjunto de dados "Sul" o MRS-*kmeans* apresentou um desempenho 15,32% superior quando comparado com o COP-*kmeans* e apenas 1,51% inferior quando comparado com o MLC-*kmeans*, considerado como uma igualdade nesse último resultado.

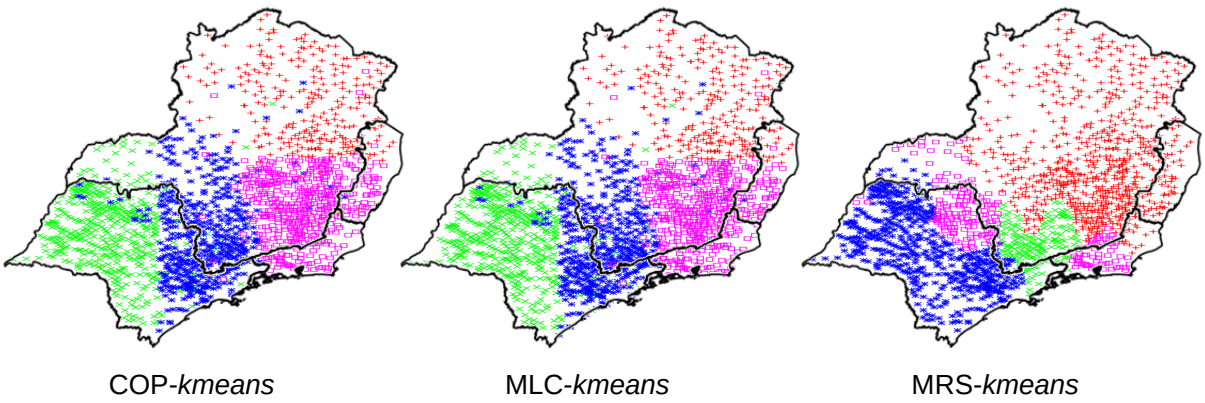

Figura 5.36 – Resultados dos experimentos na região sudeste.

Em contra partida, as regiões norte e centro-oeste tem seus municípios geograficamente mais esparsos, baseando nos dois primeiros atributos referentes a localização geográfica, o que de certa forma facilita o trabalho dos algoritmos. Para esses experimentos, no conjunto de dados "Norte" o MRS-*kmeans* apresentou um desempenho 2,5% inferior quando comparado com o COP-*kmeans* e 1,16% superior quando comparado com o MLC-*kmeans*. Já para o conjunto de dados "Centro-Oeste" o MRS-*kmeans* apresentou um desempenho 3,58% inferior quando comparado com o COP-*kmeans* e 1,66% superior quando compa-

rado com o MLC-*kmeans*. Nesses dois conjuntos pode-se perceber um certo pareamento no resultado, com uma leve superioridade do MRS-*kmeans* em relação ao MLC-*kmeans*.

A região nordeste também é difícil de se particionar, pois as cidades estão bastante concentradas mais ao litoral, além disso, os índices sócio-econômicos dos municípios litorâneos são bastante superiores ao restante dos municípios da região. Nesse conjunto de dados, "Nordeste", o MRS-*kmeans* apresentou um desempenho 0,5% inferior quando comparado com o COP-*kmeans* e 0,3% inferior quando comparado com o MLC-*kmeans*. Podendo assim observar uma equivalência no resultado dos algoritmos para esse conjunto de dados. De maneira geral, pode-se afirmar que o algoritmo MRS-*kmeans*, a partir de uma pequena quantidade de informação adicional sobre o conjunto de dados, consegue resultados satisfatórios também em conjuntos de dados reais, em comparação a algoritmos com abordagens parecidas da literatura.

5.3 Considerações Finais

O conjunto de experimentos apresentado neste capítulo mostra a eficácia e a eficiência do novo método de agrupamentos de dados por semi-supervisão desenvolvido e descrito nesta dissertação. O ponto forte do algoritmo MRS-*kmeans* é a detecção de estruturas de agrupamentos em conjuntos de dados de formas complexas, em abordagens que geram um particionamento dos dados, superando a dificuldade dessa abordagem de lidar com estruturas complexas. O algoritmo tem como grande vantagem a rápida convergência para um resultado satisfatório e assim demandando menor custo computacional. Um dos fatores para esse desempenho está relacionado ao melhor aproveitamento pelo MRS-*kmeans* das informações de semi-supervisão fornecidas pela restrições, aplicando, dessa forma, essa informação adicional a um maior número de instâncias do conjunto de dados, diferentemente dos outros algoritmos que utilizam esse conhecimento em um número restrito de instâncias. O próximo capítulo conclui esta dissertação revisando os objetivos que foram impostos a esse trabalho, além de algumas considerações e propostas para trabalhos futuros.

Capítulo 6

Conclusão

O objetivo do método de agrupamento proposto nesta dissertação é usar a informação dada a uma pequena parte do conjunto de dados, na forma de restrições, a fim de aumentar a qualidade (isto é, a interpretabilidade) do agrupamento resultante. Para isso, o método tira maior proveito das informações disponíveis em um conjunto de restrições entre pares de instâncias e incorpora esse conhecimento no processo de agrupamento, gerando assim, múltiplos representantes auxiliares para cada grupo e um novo tipo de restrições entre representantes que auxiliam na alocação desses auxiliares aos grupos mais adequado. Com isso, o algoritmo melhora a atribuição das instâncias para os grupos mais corretos, considerando a dificuldade inerente dos diferentes formatos nos conjuntos de dados.

Para realizar tal tarefa, primeiramente desenvolveu-se estratégias para transformar informações de restrições entre pares de instâncias em múltiplos representantes para cada grupo, determinando um centróide para um conjunto de instâncias interconectadas por uma ou mais restrições *must-link*. Além disso, também desenvolveu-se estratégias para derivar as informações de restrições em nível de instância para restrições em nível de protótipo, as quais auxiliam a correta atribuição dos múltiplos representantes para os grupos mais adequados. Como essas estratégias levam a um número consideravelmente grande de centróides auxiliares, surgiu a necessidade de desenvolver meios de selecionar os representantes auxiliares mais representativos, por meio da diversidade, para uma detecção de agrupamentos mais acurada e de menor custo. Uma função de distância agregada, que pode considerar vários representantes no cálculo da distância de uma instância ao grupo mais próximo, foi implantada para tirar vantagem dos centróides auxiliares e com isso detectar as formas complexas dos agrupamentos.

Ao final, uma metodologia de avaliação própria para algoritmos de detecção semi-supervisionada de agrupamentos foi utilizada para validar a superioridade do algoritmo MRS-*kmeans* em comparação com algoritmos de abordagens semelhantes da literatura. Nessa metodologia, parte das restrições são retiradas do processo de agrupamento e utilizadas para verificar as taxas de acerto no particionamento gerado pelo algoritmo, dessa forma, é possível deduzir a real eficiência das restrições no agrupamento resultante. Va-

lidações por meio testes estatísticos também ajudaram a corroborar a afirmação de bom desempenho do MRS-*kmeans*. A seguir são apresentadas as principais contribuições atingidas com essa pesquisa.

6.1 Principais Contribuições

Alicerçado pelos resultados dos experimentos realizados, as principais contribuições alcançadas no desenvolvimento do trabalho são:

- Criação de um novo método de detecção de agrupamentos semi-supervisionado por particionamento de dados. Esse novo método é capaz de aproveitar de melhor forma as informações contidas em restrições entre instâncias do tipo *must-link* e *cannot-link*, gerando um novo tipo de conhecimento utilizado para melhorar o desempenho em algoritmos com abordagens de particionamento de dados. Dentro desse novo conhecimento extraído das restrições entre instâncias estão os múltiplos centróides auxiliares, baseados inteiramente nas restrições *must-link*, e também os novos tipos de restrições que atuam em nível de protótipo, contribuindo para uma melhor distribuição dos centróides auxiliares aos grupos.

- Detecção aprimorada de agrupamentos em conjuntos de dados de estruturas complexas, possibilitando realizar processos de detecção de agrupamentos mais acurados em conjuntos de dados que possuem estruturas de agrupamentos contendo grupos de formas, tamanhos e densidades arbitrárias. A eficiência desse novo método de agrupamento é bem destacada nos resultados dos vários experimentos, onde o MRS-*kmeans* é superior na grande maioria dos resultados, mostrando ainda, sua superioridade em relação a outros algoritmos por meio de testes estatísticos.

- Realização da detecção de agrupamentos com menor número de iterações e consequentemente retornando o particionamento dos dados com um menor custo computacional. Isso é possível graças a estratégia de selecionar apenas os centróides auxiliares mais representativos para o agrupamento. Também, com a utilização de estruturas de dados que agilizam a busca por restrições durante o processo, e ainda, pela forma como são armazenadas algumas informações de distância entre os centróides auxiliares, contribuindo positivamente para um resultado com menor consumo de recursos computacionais.

6.2 Trabalhos Futuros

Embora os resultados dos experimentos mostrem o bom desempenho do novo método desenvolvido, superando algoritmos da literatura, a pesquisa desenvolvida abre espaço

para novos questionamentos e possíveis linhas para a continuação do trabalho. A seguir são listados alguns possíveis trabalhos futuros.

- Um dos pontos importantes que pretende-se investigar logo adiante é o comportamento do algoritmo MRS-*kmeans* em conjuntos de dados de alta dimensionalidade. A questão da alta dimensionalidade já um problema bastante conhecido e estudado na literatura de detecção de agrupamentos e existem várias estratégias propostas para contorna-lo. Uma possível adaptação do MRS-*kmeans* para ter resultados satisfatórios em conjuntos de alta dimensionalidade demanda um estudo mais aprofundado e poder se tornar uma interessante linha de pesquisa.

- Uma outra linha importante a se seguir está relacionada ao uso de informações adicionais em detecção de agrupamentos de imagens. O problema relacionado é que existe um *gap* semântico entre as características extraídas das imagens, as quais podem ser representadas em forma de vetores, e a percepção visual do usuário referente a uma imagem. Esse problema é conhecido como a descontinuidade semântica, e poderia ser utilizado informações de relevância do usuário como forma de semi-supervisão para detectar agrupamentos mais acurados ao que o usuário realmente deseja.

- Como apresentado na Seção 2.3.1, a função de distância agregada utilizada possui um parâmetro chamado fator de agregação que pode ser utilizado para alterar o espaço de cobertura dos representantes no momento do cálculo de distância agregada. No trabalho descrito aqui, o fator de agregação utilizado foi o menos infinito ($-\infty$), que se resume a distância do representante mais próximo de uma determinada instância. Entretanto, mudanças no fator de agregação poderiam ocasionar alterações no resultado do processo de agrupamento, e nesse caso, poderia ser melhor investigado a influência desse fator no resultado do algoritmo MRS-*kmeans*.

- O trabalho realizado, descrito nesta dissertação, teve como foco principal comparar o método desenvolvido com abordagens de detecção de agrupamento por particionamento de dados. Porém, existem outras abordagens de agrupamento descritas na literatura, que têm o objetivo de detectar agrupamentos em conjuntos de dados de estruturas complexas e seria interessante a comparação com esses outros algoritmos e detectar possíveis deficiências e vantagens de utilizar o algoritmo MRS-*kmeans*.

- Pretende-se também investigar se o desempenho do algoritmo MRS-*kmeans* se mantém em um nível razoavelmente satisfatório mesmo em grandes bases de dados. Com a popularização dos dispositivos eletrônicos e redes sociais, uma quantidade gigantesca de informações está disponível na internet, e com isso, ao longo dos anos vem sendo desenvolvidos métodos de mineração de dados para extrair informações relevantes dessas grandes massas de dados. Portanto, é interessante que métodos de

agrupamentos de dados, que são uma das formas de extração de conhecimento de conjuntos de dados, também mantenham seu desempenho em grandes bases de dados, e por isso pretende-se estudar formas da utilização e possíveis adaptações para que o algoritmo MRS-*kmeans* retorne um resultado satisfatório nesses conjuntos.

6.3 Contribuições em Produção Bibliográfica

A pesquisa envolvida nesta dissertação gerou publicações de dois artigos até o momento. A seguir são apresentados os artigos publicados em seus respectivos eventos.

1. Artigo intitulado "Usando Semi-supervisão para definir Representantes Auxiliares em Processos de Agrupamentos de Dados", apresentado no *Symposium on Knowledge Discovery, Mining and Learning* (KDMiLe 2014). Nesse artigo foram apresentados os resultados preliminares da primeira versão do algoritmo MRS-*kmeans*, o qual gerava apenas os centróides auxiliares e restrições do tipo *cannot-link* entre protótipos para auxiliar a detecção dos agrupamentos em quatro conjuntos de dados de estruturas complexas. Na ocasião, o artigo recebeu a premiação de melhor *short paper* do evento.

2. Artigo intitulado "*Semi-supervised Clustering using Multi-Assistant-Prototypes to Represent each Cluster*", apresentado no *30th ACM/SIGAPP Symposium on Applied Computing* (SAC 2015). Nesse artigo foram apresentados os resultados já com as principais caraterísticas do algoritmo MRS-*kmeans*, tal como a geração de restrições do tipo *must-link* entre protótipos e o uso da diversidade ao selecionar centróides auxiliares mais representativos. Nos experimentos desse artigo foram considerados oito conjuntos de dados de diferentes estruturas de agrupamentos.

Ainda estuda-se a possibilidade de publicação de uma versão mais estendida do trabalho realizado durante a pesquisa descrita nesta dissertação. Nesse novo artigo poderia-se explorar mais detalhes do método de agrupamento desenvolvido, como também, a apresentação dos resultados dos experimentos que testam várias dimensionalidades de um conjunto de dados.

Referências

AGRAWAL, R. et al. Automatic subspace clustering of high dimensional data for data mining applications. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 27, n. 2, p. 94–105, jun. 1998. ISSN 0163-5808.

AHMED, E. B.; NABLI, A.; GARGOURI, F. Shacun: Semi-supervised hierarchical active clustering based on ranking constraints. In: **Proceedings of the 12th Industrial Conference on Advances in Data Mining: Applications and Theoretical Aspects**. Berlin, Heidelberg: Springer-Verlag, 2012. (ICDM'12), p. 194–208.

_____. A new semi-supervised hierarchical active clustering based on ranking constraints for analysts groupization. **Appl. Intell.**, v. 39, n. 2, p. 236–250, 2013.

ALBUQUERQUE, G.; LÖWE, T.; MAGNOR, M. Synthetic generation of high-dimensional datasets. **IEEE Transactions on Visualization and Computer Graphics (TVCG, Proc. Visualization / InfoVis)**, v. 17, n. 12, p. 2317–2324, dez. 2011.

ANKERST, M. et al. Optics: Ordering points to identify the clustering structure. In: **Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1999. (SIGMOD '99), p. 49–60.

BARBARA, D. **An Introduction to Cluster Analysis for Data Mining**. [S.l.]: http://www.cs.wustl.edu/žhang/teaching/cs514/fall13/Intro-Clustering.pdf, 2000.

BARIONI, M. C. N. et al. Open issues for partitioning clustering methods: an overview. **Wiley Interdisc. Rew.: DMKD**, v. 4, n. 3, p. 161–177, 2014.

BASU, S.; DAVIDSON, I.; WAGSTAFF, K. **Constrained Clustering: Advances in Algorithms, Theory, and Applications**. 1. ed. [S.l.]: Chapman & Hall/CRC, 2008.

BILENKO, M.; BASU, S.; MOONEY, R. J. Integrating constraints and metric learning in semi-supervised clustering. In: **Proceedings of the Twenty-first International Conference on Machine Learning**. New York, NY, USA: ACM, 2004. (ICML '04), p. 11.

CARVALHO, A. X. Y. et al. **Dinâmica dos Municípios**. [S.l.]: IPEA, 2008.

CHAPELLE, O.; SCHÖLKOPF, B.; ZIEN, A. (Ed.). **Semi-Supervised Learning**. Cambridge, MA: MIT Press, 2006.

COLLINS, J.; OKADA, K. A comparative study of similarity measures for content-based medical image retrieval. In: **CLEF (Online Working Notes/Labs/Workshop)**. [S.l.: s.n.], 2012.

DAVIDSON, I.; BASU, S. A survey of clustering instance level. In: **Transactions on Knowledge Discovery from Data**. [S.l.]: ACM, 2007. p. 1–41.

DBLP. **DBLP Computer Science Bibliography**. 2015. Http://dblp.uni-trier.de/. Acessado em: 2015-04-16.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **J. Mach. Learn. Res.**, v. 7, p. 1–30, 2006.

DISTRIBUTIONPAINTER. **Scalable Visual Analytics**. 2014. Http://www.cg.cs.tu-bs.de/projects/scalable-visual-analytics/DistributionPainter_Linux_v081_x86.zip. Acessado em: 2014-11-27.

DUBEY, A.; BHATTACHARYA, I.; GODBOLE, S. A cluster-level semi-supervision model for interactive clustering. In: **Proceedings of the 2010 European conference on Machine learning and knowledge discovery in databases: Part I**. Berlin, Heidelberg: Springer-Verlag, 2010. (ECML PKDD'10), p. 409–424.

DUNN, J.; DUNN, O. J. Multiple comparisons among means. **American Statistical Association**, p. 52–64, 1961.

ESTER, M. et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: SIMOUDIS, E.; HAN, J.; FAYYAD, U. M. (Ed.). **KDD**. [S.l.]: AAAI Press, 1996. p. 226–231.

FACELI, K. et al. **Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina**. 1. ed. [S.l.]: LTC, 2011.

FRIEDMAN, M. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. **Journal of the American Statistical Association**, American Statistical Association, v. 32, n. 200, p. 675–701, dez. 1937.

GAN, G.; MA, C.; WU, J. **Data Clustering: Theory, Algorithms and Applications**. [S.l.]: Asa-Siam, 2007.

GIONIS, A.; MANNILA, H.; TSAPARAS, P. Clustering aggregation. **ACM TKDD**, ACM, v. 1, n. 1, 2007. ISSN 1556–4681.

GUHA, S.; RASTOGI, R.; SHIM, K. Cure: An efficient clustering algorithm for large databases. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 27, n. 2, p. 73–84, jun. 1998.

_____. Rock: A robust clustering algorithm for categorical attributes. In: **In Proc.ofthe15thInt.Conf.onDataEngineering**. [S.l.: s.n.], 2000.

HALKIDI, M.; BATISTAKIS, Y.; VAZIRGIANNIS, M. Cluster validity methods: Part i. **SIGMOD Rec.**, ACM, New York, NY, USA, v. 31, n. 2, p. 40–45, jun. 2002.

HAN, J.; KAMBER, M. **Data Mining: Concepts and Techniques**. 2. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

HAN, J.; KAMBER, M.; PEI, J. **Data Mining: Concepts and Techniques**. 3. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

HINNEBURG, A.; KEIM, D. A. An efficient approach to clustering in large multimedia databases with noise. In: AGRAWAL, R.; STOLORZ, P. E.; PIATETSKY-SHAPIRO, G. (Ed.). **KDD**. [S.l.]: AAAI Press, 1998. p. 58–65.

_____. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In: **Proceedings of the 25th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999. (VLDB '99), p. 506–517. ISBN 1-55860-615-7.

HUANG, H.; CHENG, Y.; ZHAO, R. A semi-supervised clustering algorithm based on must-link set. In: **Proceedings of the 4th international conference on Advanced Data Mining and Applications**. Berlin, Heidelberg: Springer-Verlag, 2008. (ADMA '08), p. 492–499.

IMAN, R. L.; DAVENPORT, J. M. Approximations of the critical region of the Friedman statistic. **Communications in Statistics - Theory and Methods**, Taylor & Francis, v. 9, n. 6, p. 571–595, jan. 1980.

IPEA. **Instituto de Pesquisa Econômica Aplicada**. 2015. Http://www.ipeadata.gov.br/. Acessado em: 2015-01-10.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern Recogn. Lett.**, Elsevier Science Inc., New York, NY, USA, v. 31, n. 8, p. 651–666, jun. 2010.

JIANG, H. et al. Extracting elite pairwise constraints for clustering. **Neurocomput.**, Elsevier Science Publishers B. V., Amsterdam, Netherlands, v. 99, p. 124–133, jan. 2013.

KAUFMAN, L.; ROUSSEEUW, P. J. **Finding groups in data: an introduction to cluster analysis**. New York: John Wiley and Sons, 1990.

KUMAR, N.; KUMMAMURU, K. Semisupervised clustering with metric learning using relative comparisons. **IEEE Trans. on Knowl. and Data Eng.**, IEEE Educational Activities Department, Piscataway, NJ, USA, v. 20, n. 4, p. 496–503, abr. 2008.

KUTZ, O. et al. Logics of metric spaces. **ACM Transactions on Computational Logic**, ACM, New York, NY, USA, v. 4, n. 2, p. 260–294, april 2003. ISSN 1529-3785.

LAI, H. P. et al. A new interactive semi-supervised clustering model for large image database indexing. **Pattern Recognition Letters**, v. 37, p. 94–106, 2014.

LIU, E. Y.; ZHANG, Z.; WANG, W. Clustering with relative constraints. In: **Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2011. (KDD '11), p. 947–955.

LIU, M.; JIANG, X.; KOT, A. C. A multi-prototype clustering algorithm. **Pattern Recognition**, v. 42, n. 5, p. 689–698, 2009.

LIU, Y.; JIN, R.; JAIN, A. K. Boostcluster: Boosting clustering by pairwise constraints. In: **Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. New York, NY, USA: ACM, 2007. (KDD '07), p. 450–459.

LUO, T. et al. A multi-prototype clustering algorithm based on minimum spanning tree. In: **Seventh International FSKD - Conference on Fuzzy Systems and Knowledge Discovery**. Yantai, Shandong, China: [s.n.], 2010. p. 1602–1607.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: **In 5-th Berkeley Symposium on Mathematical Statistics and Probability**. [S.l.: s.n.], 1967. p. 281–297.

MUELLER, M.; KRAMER, S. Integer linear programming models for constrained clustering. In: **Proceedings of the 13th International Conference on Discovery Science**. Berlin, Heidelberg: Springer-Verlag, 2010. (DS'10), p. 159–173.

NAGESH, H.; GOIL, S.; CHOUDHARY, A. Adaptive grids for clustering massive data sets. **SIAM Conference on Data Mining**, 2001.

NETTO, P. O. B. **Grafos: Teoria, Modelos, Algoritmos**. 4 ed.. ed. São Paulo, SP: Edgard Blucher, 2006.

NG, R. T.; HAN, J. Efficient and effective clustering methods for spatial data mining. In: **Proceedings of the 20th International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994. (VLDB '94), p. 144–155.

PATERLINI, A. A.; NASCIMENTO, M. A.; JR., C. T. Using pivots to speed-up k-medoids clustering. **JIDM**, v. 2, n. 2, p. 221, 2011.

PENSA, R. G. et al. Co-clustering numerical data under user-defined constraints. **Statistical Analysis and Data Mining**, v. 3, n. 1, p. 38–55, 2010.

POURRAJABI, M. et al. Model selection for semi-supervised clustering. In: **Proc. 17th International Conference on Extending Database Technology (EDBT)**. Athens, Greece: [s.n.], 2014. p. 331–342.

RAND, W. Objective criteria for the evaluation of clustering methods. **Journal of the American Statistical Association**, v. 66, n. 336, p. 846–850, 1971.

RAZENTE, H. L. **Adequando consultas por similaridade para reduzir a descontinuidade semântica na recuperação de imagens por conteúdo**. Dissertação (Tese de Doutorado) — Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2009.

RAZENTE, H. L. et al. Aggregate similarity queries in relevance feedback methods for content-based image retrieval. In: **SAC**. [S.l.: s.n.], 2008. p. 869–874.

SAAD, S. M.; KAMARUDIN, S. S. Comparative analysis of similarity measures for sentence level semantic measurement of text. In: **ICCSCE**. [S.l.: s.n.], 2013. p. 90–94.

SCHMIDT, J.; BRANDLE, E. M.; KRAMER, S. Clustering with attribute-level constraints. In: **Proceedings of the 2011 IEEE 11th International Conference on Data Mining**. Washington, DC, USA: IEEE Computer Society, 2011. (ICDM '11), p. 1206–1211.

SHEIKHOLESLAMI, G.; CHATTERJEE, S.; ZHANG, A. Wavecluster: A multi-resolution clustering approach for very large spatial databases. In: GUPTA, A.; SHMUELI, O.; WIDOM, J. (Ed.). **VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases**. [S.l.]: Morgan Kaufmann, 1998. p. 428–439.

SLIMANI, T. Description and evaluation of semantic similarity measures approaches. **CoRR**, abs/1310.8059, 2013.

TIBSHIRANI, R.; GUENTHER, W.; HASTIE., T. Estimating the number of clusters in a data set via the gap statistic. 2001.

VENDRAMIN, L.; CAMPELLO, R. J. G. B.; HRUSCHKA, E. R. Relative clustering validity criteria: A comparative overview. **Stat. Anal. Data Min.**, John Wiley & Sons, Inc., New York, NY, USA, v. 3, n. 4, p. 209–235, ago. 2010.

WAGSTAFF, K.; CARDIE, C. Clustering with instance-level constraints. In: **Proceedings of the Seventeenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000. (ICML '00), p. 1103–1110.

WAGSTAFF, K. et al. Constrained k-means clustering with background knowledge. In: **Proceedings of the Eighteenth International Conference on Machine Learning**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001. (ICML '01), p. 577–584.

WANG, W.; YANG, J.; MUNTZ, R. R. Sting: A statistical information grid approach to spatial data mining. In: **Proceedings of the 23rd International Conference on Very Large Data Bases**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. (VLDB '97), p. 186–195. ISBN 1-55860-470-7.

WITTEN, I. H.; FRANK, E. **Data Mining: Practical Machine Learning Tools and Techniques**. 2nd. ed. [S.l.]: Morgan Kaufmann, 2005.

ZAKI, M. J.; MEIRA, W. **Data Mining and Analysis: Fundamental Concepts and Algorithms**. [S.l.]: Cambridge University Press, 2014.

ZAR, J. H. **Biostatistical Analysis**. 5th. ed. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2007.

ZHANG, T.; RAMAKRISHNAN, R.; LIVNY, M. Birch: An efficient data clustering method for very large databases. In: **Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data**. New York, NY, USA: ACM, 1996. (SIGMOD '96), p. 103–114.

ZHENG, L.; LI, T. Semi-supervised hierarchical clustering. In: **ICDM**. Vancouver: IEEE, 2011. p. 982–991.

Apêndices

APÊNDICE **A**

Tabelas Estatísticas

Nas tabelas estatísticas apresentadas a seguir são descritos os valores críticos necessários para a validação estatística. Cada valor representa um nível de confiança para o teste, considerando também o grau de liberdade imposto pelo problema.

A.1 Valores Críticos da F-*Distribution*

A seguir são apresentadas as tabelas com os valores críticos de acordo com a F-*Distribution*, mais informações sobre essas tabelas podem ser encontradas em (ZAR, 2007). A Tabela 1.4 apresenta os valores críticos considerando o nível de confiança $\alpha = 0.05$, o mesmo que 95% de confiança. Já a Tabela 1.5 apresenta os valores críticos considerando o nível de confiança $\alpha = 0.10$, ou seja, 90% de confiança.

Tabela 1.4 – Valores críticos da F-*Distribution* para $\alpha = 0.05$.

$\alpha = 0.05$	$DF_1 = 1$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$DF_2 = 1$	161,44	199,50	215,70	224,58	230,16	233,98	236,76	238,88	240,54	241,88
2	18,51	19	19,16	19,24	19,29	19,32	19,35	19,37	19,38	19,39
3	10,12	9,55	9,27	9,11	9,01	8,94	8,88	8,84	8,81	8,78
4	7,70	6,94	6,59	6,38	6,25	6,16	6,09	6,04	5,99	5,96
5	6,60	5,78	5,40	5,19	5,05	4,95	4,87	4,81	4,77	4,73
6	5,98	5,14	4,75	4,53	4,38	4,28	4,20	4,14	4,09	4,06
7	5,59	4,73	4,34	4,12	3,97	3,86	3,78	3,72	3,67	3,63
8	5,31	4,45	4,06	3,83	3,68	3,58	3,50	3,43	3,38	3,34
9	5,11	4,25	3,86	3,63	3,48	3,37	3,29	3,22	3,17	3,13
10	4,96	4,10	3,70	3,47	3,32	3,21	3,13	3,07	3,02	2,97
20	4,35	3,49	3,09	2,86	2,71	2,59	2,51	2,44	2,39	2,34
30	4,17	3,31	2,92	2,68	2,53	2,42	2,33	2,26	2,21	2,16
40	4,08	3,23	2,83	2,60	2,44	2,33	2,24	2,18	2,12	2,07
50	4,03	3,18	2,79	2,55	2,40	2,28	2,19	2,12	2,07	2,02
78	3,96	3,11	2,72	2,48	2,33	2,21	2,12	2,05	2,00	1,95

Tabela 1.5 – Valores críticos da F-*Distribution* para $\alpha = 0.10$.

$\alpha = 0.10$	$DF_1 = 1$	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$DF_2 = 1$	39,86	49,50	53,59	55,83	57,24	58,20	58,90	59,43	59,85	60,19
2	8,52	9	9,16	9,24	9,29	9,32	9,34	9,36	9,38	9,39
3	5,53	5,46	5,39	5,34	5,30	5,28	5,26	5,25	5,24	5,23
4	4,54	4,32	4,19	4,10	4,05	4	3,97	3,95	3,93	3,91
5	4,06	3,77	3,61	3,52	3,45	3,40	3,36	3,33	3,31	3,29
6	3,77	3,46	3,28	3,18	3,10	3,05	3,01	2,98	2,95	2,93
7	3,58	3,25	3,07	2,96	2,88	2,82	2,78	2,75	2,72	2,70
8	3,54	3,11	2,92	2,80	2,72	2,66	2,62	2,58	2,56	2,53
9	3,36	3	2,81	2,69	2,61	2,55	2,50	2,46	2,44	2,41
10	3,28	2,92	2,72	2,60	2,52	2,46	2,41	2,37	2,34	2,32
20	2,97	2,58	2,38	2,24	2,15	2,09	2,03	1,99	1,96	1,93
30	2,88	2,48	2,27	2,14	2,04	1,98	1,92	1,88	1,84	1,81
40	2,83	2,44	2,22	2,09	1,99	1,92	1,87	1,82	1,79	1,76
50	2,80	2,41	2,19	2,06	1,96	1,89	1,84	1,79	1,75	1,72
78	2,77	2,37	2,15	2,01	1,92	1,85	1,79	1,75	1,71	1,68

A.2 Valores Críticos do Teste *Bonferroni-Dunn*

A Tabela 1.6 apresentada logo a seguir descreve os valores críticos para o teste *post-hoc Bonferroni-Dunn*, aplicado após o teste de *Friedman*. Os valores são baseados na *Studentized Range Statistic* divida por $\sqrt{2}$ (DEMSAR, 2006), considerando o respectivo nível de confiança q_α. O número de algoritmos a se considerar deve incluir o algoritmo de controle.

Tabela 1.6 – Valores críticos para o teste *Bonferroni-Dunn*.

#algoritmos	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$q_{0,05}$	1,960	2,241	2,394	2,498	2,576	2,638	2,690	2,724	2,773
$q_{0,10}$	1,645	1,960	2,128	2,241	2,326	2,394	2,450	2,498	2,539