Favorecendo o Desempenho do k-Means via Métodos de Inicialização de Centroides

Anderson Francisco de Oliveira

Agosto / 2018

Dissertação de Mestrado em Ciência da Computação

Favorecendo o Desempenho do k-Means via Métodos de Inicialização de Centroides

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

Campo Limpo Paulista, 7 de Agosto de 2018.

Anderson Francisco de Oliveira

Profa. Dra. Maria do Carmo Nicoletti (Orientadora)

Ficha catalográfica elaborada pela

Biblioteca Central da UNIFACCAMP

O45F

Oliveira, Anderson Francisco de

Favorecendo o desempenho do k-means via métodos de inicialização de centroides / Anderson Francisco de Oliveira. Campo Limpo Paulista, SP: UNIFACCAMP, 2018.

Orientadora: Profª. Drª. Maria do Carmo Nicoletti

Dissertação (Programa de Mestrado Profissional em Ciência da Computação) – Centro Universitário Campo Limpo Paulista – UNIFACCAMP.

1. Aprendizado de máquina. 2. Aprendizado não supervisionado. 3. K-means. 4. Algoritmos de inicialização. I. Nicoletti, Maria do Carmo. II. Centro Universitário Campo Limpo Paulista. III. Título.

CDD-005.1

Resumo. *Agrupamento pode ser estabelecido de uma maneira simplista como: dado um conjunto de padrões X, encontrar a melhor forma de dividi-los em grupos disjuntos de padrões, de maneira que a união de tais grupos recomponha X. A categoria mais simples de algoritmos de agrupamento é a particional. Algoritmos particionais organizam os padrões de dados de um conjunto em vários grupos disjuntos. O algoritmo k-Means é um dos mais conhecidos e comumente usados dos métodos particionais. No entanto, a sua inicialização pode impactar negativamente o agrupamento produzido pelo algoritmo. Este projeto investiga alguns métodos de inicialização propostos na literatura, com o objetivo de melhorar o desempenho do algoritmo k-Means, por meio do uso de uma inicialização mais eficiente induzindo melhores agrupamentos e diminuindo o número de iterações para o k-Means convergir.*

Palavras-chave*: aprendizado de máquina, aprendizado não supervisionado, k-Means, algoritmos de inicialização.*

Abstract. *Clustering can be stated in a simplistic way as: given a set of pattern X, find the best way to divide them into disjoint groups of patterns, so that their union restores X. The simplest category of clustering algorithms is the partitional. Partitional algorithms organize the data patterns in a clustering of disjoint clusters. The k-Means algorithm is one of the most popular and well-known partitional algorithm. However, its initialization step can negative impact on the produced clustering. This project investigates a few initialization methods proposed in the literature, aiming at improving the performance of the k-Means algorithm by using a more efficient initialization inducing better clusters and decreasing the number of iterations for k-Means to converge.*

Keywords: *machine learning, unsupervised learning, k-Means, initialization algorithms.*

Dedicatória

Dedico essa dissertação à minha esposa Tatiane, filhos Gabriel e Lorena, meu incentivo para continuar se renova a cada dia pelo amor que tenho por vocês, sozinho essa caminhada não seria possível, essa projeto é nosso.

Em especial meu Pai João, mãe Maria Lúcia e irmãos que sempre me apoiaram e deram forças para continuar.

Agradecimentos

Agradeço primeiramente a Deus por me conduzir, sustentar e dar sabedoria para realização deste projeto.

Aos diretores da empresa que trabalho, Aderbal e Guilherme pelo incentivo e apoio durante todo o percurso.

À minha orientadora professora Dra. Maria do Carmo Nicoletti pela dedicação, paciência, motivação e ensinamentos durante todo o processo de desenvolvimento desta dissertação.

Aos professores Dr. Osvaldo Luiz de Oliveira, José Hiroki Saito pela disposição e contribuições ao trabalho examinado.

Aos amigos Vitor Martins, Bruno Ponsoni, Vagner Scamati, Bruno Amaral e Heber Miranda que contribuíram durante o percurso.

Minha gratidão também ao Pr. Ezequiel e Pra. Amanda pelas palavras de incentivos e orações.

Aos professores e funcionários do programa de mestrado em Ciência da Computação do Centro Universitário Campo Limpo Paulista.

A todos que contribuíram para a realização deste trabalho.

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

Capítulo 1

Introdução

Um problema recorrente em uma grande variedade de áreas de pesquisa tais como reconhecimento de padrões, aprendizado de máquina, mineração de dados e estatística, entre outras, é caracterizado como *problema de agrupamento*. Tal problema pode ser descrito de maneira simplista como: dado um conjunto de dados (observações, objetos, pontos, etc.), agrupar dados semelhantes em grupos.

Um agrupamento de um determinado conjunto de dados, então, é caracterizado como um conjunto de grupos, em que elementos pertencentes a um grupo são similares entre si e elementos pertencentes a grupos distintos não são similares. A solução para o problema de agrupamento pode ser abordada de várias maneiras; uma delas é conhecida como abordagem particional, em que o problema de agrupamento é visto como um problema de partição do conjunto de dados.

Dentre os vários algoritmos particionais existentes na literatura, o algoritmo conhecido como k-Means [MacQueen 1967] é considerado o mais popular e um dos que teve maior sucesso em um grande número de aplicações. Um aspecto particular do k-Means é o objeto de pesquisa deste trabalho e abordado no que segue.

1.1 Motivação e Objetivo

Como comentado anteriormente, o k-Means é um dos algoritmos mais utilizados na busca para a solução do problema de agrupamento. É sabido, entretanto, que o k-Means sofre de um problema conhecido como problemas de inicialização, relacionado ao agrupamento inicial, a partir do qual o processo iterativo conduzido pelo algoritmo começa. Na literatura podem ser encontrados vários algoritmos que buscam resolver o problema da inicialização dos grupos, com o objetivo de melhorar a qualidade dos agrupamentos induzidos pelo k-

Means. A pesquisa descrita nesta dissertação investigou alguns desses algoritmos, com o objetivo de identificar suas reais contribuições ao problema, avaliar quão factíveis suas implementações são, examinar a efetiva originalidade de cada um deles e, também, compará-los em relação aos resultados obtidos.

Como parte do trabalho de pesquisa realizado, foi projetado e programado um sistema computacional que agrega tanto a implementação do k-Means quanto ao dos vários algoritmos de inicialização de centros de grupos que foram investigados, com vistas a disponibilizar um ambiente voltado à experimentação, aprendizado e ensino.

1.2 Organização da Dissertação

Este capítulo inicial, além de contextualizar a pesquisa realizada e discutir algumas das motivações que subsidiaram a sua proposta, apresenta a organização desta dissertação, descrevendo brevemente e de maneira sequencial, o conteúdo dos demais capítulos que a compõem, como segue.

Capítulo 2: Introduz a área de pesquisa de Aprendizado Indutivo de Máquina (AIM) ou simplesmente Aprendizado de Máquina (AM), uma vez que o problema de agrupamento pode ser tratado por uma dentre as várias abordagens de AM *i.e.*, aquela implementada pelos chamados *algoritmos não supervisionados*.

No capítulo, inicialmente, a área de AM é tratada de maneira geral, em que são apresentadas definições, principais formas de AM e caracterização de conjuntos de treinamento, teste e validação, entre outros. O conteúdo do capítulo foi construído com o objetivo de fornecer um contexto de conceituações básicas relacionadas à área de AM, que subsidiaram o trabalho realizado.

Capítulo 3: Tem por foco a caracterização detalhada de algoritmos de aprendizado não supervisionado, particularmente os algoritmos de agrupamento. O capítulo aborda a notação empregada e as definições básicas utilizadas no texto. Também, discute um conceito altamente relevante para tais algoritmos, aquele da medida de similaridade, muitas vezes consideradas como distâncias. A última seção do capítulo aborda detalhadamente o

algoritmo que foi alvo desta pesquisa *i.e.*, o k-Means e nela, são também apresentadas descrições dos índices de validação utilizados nos experimentos reportados no Capítulo 6.

Capítulo 4: Optou-se pela apresentação dos algoritmos de inicialização investigados no trabalho de pesquisa descrito nesta dissertação em dois capítulos, de maneira a balancear o volume de informações.

Este capítulo, particularmente, apresenta e discute dois algoritmos propostos na literatura com o objetivo de colaborar com o algoritmo k-Means, por meio de um processo de inicialização mais refinado. A Seção 4.2 discute em detalhes o algoritmo k-Means++ [Arthur & Vassilvitskii 2007] e a Seção 4.3, o algoritmo SPSS (*Single Pass Seed Selection*) [Pavan et *al.* 2010] [Pavan et *al.* 2011].

Capítulo 5: Aborda três outros algoritmos propostos na literatura, com o intuito de otimizar a fase de inicialização do k-Means. São eles o CCIA (*Cluster Center Initialization Algorithm*) [Khan & Ahmad 2004], o algoritmo *Method1* [Al-Daoud & Roberts 1996] e o algoritmo Maedeh-Suresh [Maedeh & Suresh 2013].

Capítulo 6: O capítulo inicialmente apresenta e descreve as principais características e funcionalidades de um ambiente computacional que foi projetado e desenvolvido com vistas à experimentação.

Na sequência descreve o conjunto de experimentos realizados utilizando os algoritmos de inicialização investigados, apresentando os conjuntos de dados escolhidos, a metodologia de experimentação adotada, os resultados obtidos, bem como uma discussão sobre os resultados, feita com foco na comparação entre eles. A seção que finaliza o capítulo resume as principais conclusões sobre o trabalho realizado, elenca alguns dos problemas enfrentados e discute como alguns deles foram superados. Um diagrama com a estruturação dos algoritmos investigados (e respectivos capítulos/seções em que são tratados), está apresentado na Figura 1.1.

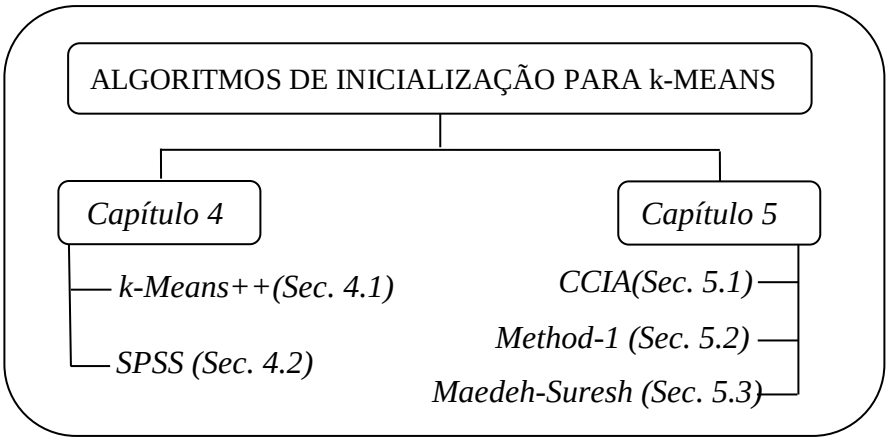


Figura 1.1 Organização da apresentação e discussão das principais características de cada um dos cinco algoritmos investigados.

Devido ao fato de alguns algoritmos investigados terem sido apresentados por meio de uma descrição ambígua e/ou incompleta nos trabalhos em que foram publicados, optou-se por manter as descrições originais de todos os algoritmos, como foram publicadas. A justificativa para essa decisão foi a de colaborar com a pesquisa na área,

(1) evidenciando os problemas de indefinição no pseudocódigo dos algoritmos publicados;

(2) evidenciando situações de exceção que o algoritmo abordado não prevê e, particularmente, que surgem como consequência de uma definição descuidada;

(3) discutindo o quanto uma abordagem descuidada e não rigorosa pode impactar no entendimento/implementação de novos algoritmos propostos e contribuir, de certa forma, para a propagação de determinados erros.

Capítulo 2

Aprendizado Indutivo de Máquina: Características e Objetivos

Este capítulo contextualiza a área de pesquisa de aprendizado indutivo de máquina, na qual esse projeto de pesquisa se insere. Para tanto, apresenta uma visão geral da área, discute alguns de seus principais objetivos e brevemente apresenta alguns conceitos relevantes envolvidos.

2.1 Aprendizado Indutivo de Máquina

O chamado Aprendizado Indutivo de Máquina é o modelo de AM mais bem sucedido e, talvez por essa razão, o mais popular; via de regra, ao se tratar com algoritmos de AM, assume-se que os algoritmos em questão são algoritmos indutivos. AM pode ser implementado por meio de inúmeros algoritmos subsidiados por uma grande variedade de formalismos matemáticos/lógicos. As referências [Mitchell 1997] [Duda *et al.*, 2001] [Witten *et al.* 2011] e [Han *et al.* 2012], entre muitas outras, apresentam e discutem um grande número deles.

Uma maneira simplista de abordar AM é como um processo com duas fases, descrito por um algoritmo, cujo resultado final é, muitas vezes, a geração de um classificador. As duas fases são:

(1) *treinamento*: em que, a partir de um conjunto de situações concretas, nomeadas de dados (também referenciados como instâncias ou exemplos), que representam um determinado conceito, o algoritmo produz, via de regra por meio de um processo indutivo, uma descrição geral do conceito que tais dados representam. Tal conjunto é chamado de *conjunto de treinamento* e cada dado de tal conjunto é chamado de *instância de treinamento*. A descrição do conceito produzida é, geralmente, chamada de *classificador*, e pode ser

representada utilizando várias estruturas de dados (árvore de decisão, regras, redes, etc.), na dependência do algoritmo utilizado.

(2) *classificação*: em que a descrição geral do conceito aprendida na fase (1) de treinamento (*i.e.*, o classificador) é utilizada para identificar novas instâncias como sendo (ou não) representantes do conceito em questão.

O processo de indução realizado por um algoritmo de AM pode ser tratado como uma busca em um espaço de hipóteses, com vistas a encontrar aquela(s) que 'melhor' representa(m) os dados do conjunto de treinamento. Nesse contexto, 'melhor' pode ser definido em termos de certos critérios como, por exemplo, precisão, compreensibilidade ou simplicidade [Nicoletti 1994].

Algoritmos que viabilizam AM têm inúmeras aplicações em áreas tais como processamento de linguagem natural, mineração de dados, diagnósticos, análises de mercado de ações, visão computacional, recuperação de informações e em muitas outras mais.

2.2 Aprendizado Supervisionado, Não Supervisionado e Semissupervisionado

Como existem inúmeros algoritmos de AM, fazendo uso dos mais variados formalismos matemáticos, pesquisadores da área têm investido, também, no desenvolvimento de taxonomias, com vistas a organizar tais algoritmos, de maneira a agrupá-los de acordo com determinados critérios.

Na literatura podem ser encontradas várias propostas de taxonomias (ver, por exemplo, a descrita em [Mitchell 1997]) e, dentre elas, aquela que adota como critério de organização, o nível da supervisão (externa) associada ao conjunto de treinamento, durante a fase de aprendizado, taxonomia que é brevemente abordada nessa seção. Como comentado anteriormente, para viabilizar a fase de treinamento de um algoritmo de AM é necessário que um conjunto de situações concretas (*conjunto de treinamento*) que representam um determinado conceito, seja fornecido ao algoritmo.

A dependência do conjunto de treinamento para que um algoritmo possa viabilizar o aprendizado é o que caracteriza o algoritmo como indutivo: a partir de um conjunto de situações concretas (conjunto de treinamento), o algoritmo (*i.e.*, o software que o implementa) gera um conjunto de expressões (regras, árvores de decisão, etc...) que

generalizam o conjunto de treinamento. As expressões geradas podem, então, ser usadas a seguir, para classificar novos dados.

As linguagens de descrição, tanto de conjuntos de treinamento quanto de expressões de conceitos induzidas por algoritmos de AM, variam bastante; diferentes algoritmos esperam que o conjunto de treinamento esteja descrito de acordo com a sintaxe de uma determinada linguagem (ver [Nicoletti 1994]).

Dentre as linguagens de descrição dos dados de treinamento mais comuns, encontra-se aquela baseada em um conjunto de atributos. Quando do uso de tal linguagem e considerando um conjunto de atributos, uma instância de dado é descrita por um conjunto de valores, em que cada valor está associado a um dos atributos do conjunto e, dependendo da situação, também pelo valor de uma *classe* associada à instância, indicando qual conceito a instância representa. A classe pode ser tratada como um tipo particular de atributo, que pode (ou não) participar da descrição de cada uma das instâncias de dados do conjunto de treinamento. O valor associado à classe é, na maioria dos casos, determinada por um especialista humano da área de conhecimento descrita pelos dados.

A taxonomia subsidiada pelo critério baseado no nível de supervisão requerido pelo algoritmo, agrupa algoritmos de AM em três diferentes grupos, na dependência do nível de supervisão requerido pelo algoritmo para realizar o aprendizado: (1) *aprendizado supervisionado*, (2) *aprendizado não supervisionado* e (3) *aprendizado semi-supervisionado*, que são brevemente caracterizados a seguir.

O fato da classe participar da descrição dos dados e, também, ser usada pelo algoritmo de aprendizado para a indução da expressão do conceito, caracteriza tal algoritmo como um *algoritmo de aprendizado supervisionado*, no sentido que o algoritmo 'precisa' de uma informação externa (no caso, a informação identificada como classe, fornecida por algo ou alguém), para poder generalizar o conjunto de dados que recebe como entrada.

Algoritmos supervisionados são os mais numerosos e podem empregar uma diversidade bem grande de formalismos, para a realização do aprendizado e representação do conceito aprendido. Dentre os algoritmos supervisionados mais populares destacam-se: (1) *simbólicos*, tais como: CN2 [Clark & Niblett 1989], ID3 [Quinlan 1986], C4.5 [Quinlan 1993], (2) *baseados em vizinhos mais próximos*, tal como o kNN [Altman 1992] e os (3) *neurais*, tais como: Backpropagation [Bishop 2006], [Gallant 1994] e, dentre as redes, as caracterizadas como construtivas [Nicoletti

et al., 2010], induzidas por algoritmos construtivos como o Tower e o Pyramid [Gallant 1990], o BabCoNN [Bertini & Nicoletti 2008] e vários outros.

Os algoritmos caracterizados como de *aprendizado não supervisionado* não fazem uso da informação dada pela classe à qual cada uma das instâncias de treinamento pertence (mesmo que a classe de cada instância de treinamento faça parte de sua descrição). Tais algoritmos geralmente aprendem por meio da identificação de subconjuntos de dados que compartilham certas similaridades. Os chamados algoritmos de agrupamento (abordados no Capítulo 3, de maneira geral, e no Capítulo 4, com foco naquele identificado como *k-Means* [MacQueen 1967] e objeto de pesquisa deste projeto) são os que mais precisamente caracterizam esse grupo. Alguns tipos de redes neurais, tais como as de Hebb e de Kohonen (ver [Bishop 2006]), podem também pertencer a esse grupo.

Algoritmos que implementam *aprendizado semi-supervisionado* têm aplicabilidade, principalmente, em situações de aprendizado automático em que o conjunto de treinamento é constituído por: (a) um subconjunto (geralmente pequeno) de instâncias de treinamento que incorporam em suas respectivas descrições, a classe e (b) por um segundo subconjunto contendo instâncias de treinamento que não incorporam a informação da classe. Estratégias que implementam o aprendizado semi-supervisionado geralmente empregam um algoritmo de AM supervisionado para induzir um classificador, usando as instâncias de treinamento agrupadas no conjunto e, então, (a) utilizam tal classificador para classificar as instâncias de treinamento contidas no conjunto (b) e, ciclicamente voltam a repetir o processo.

2.3 Os Conjuntos de Treinamento, Teste e Validação em um Contexto de Aprendizado Supervisionado

Como brevemente mencionado na Seção 2.1, para viabilizar a fase de treinamento de um algoritmo de AM é imperativo que um conjunto de dados, conhecido como *conjunto de treinamento*, que representa instâncias concretas dos conceitos a serem aprendidos, esteja disponível.

A Figura 2.1 exibe um esquema básico de AM, no qual um algoritmo de aprendizado (implementado como um software que é executado em um determinado ambiente computacional) induz um classificador (também referenciado como modelo), a partir de um conjunto de treinamento fornecido como entrada.

Nesse caso específico, como a informação sobre a classe associada a cada dado (*i.e.*, o último valor na linha que o descreve, no *conjunto de treinamento* mostrado na Figura 2.1) é utilizada pelo algoritmo de AM, o aprendizado é caracterizado como *supervisionado* e seu resultado, referenciado como *classificador*, é descrito por uma árvore de decisão no caso específico do exemplo mostrado na Figura 2.1. Uma vez induzido, o classificador pode então ser usado para classificar novos dados (de classe desconhecida).

No exemplo mostrado na Figura 2.1 cada instância de treinamento do conjunto de treinamento é descrita por um vetor com dez valores, sendo que os nove primeiros representam valores de nove atributos e o décimo valor, a identificação da classe à qual cada uma das instâncias pertence. A segunda instância do conjunto de treinamento (em negrito na figura), por exemplo, é representada pelo vetor

1,523 13,31 3,58 0,82 71,99 0,12 10,17 0,00 0,03 1

em que os valores para os atributos A_1 = 1,523, A_2 = 13,31, ..., A_9 = 0,03 e classe = 1.

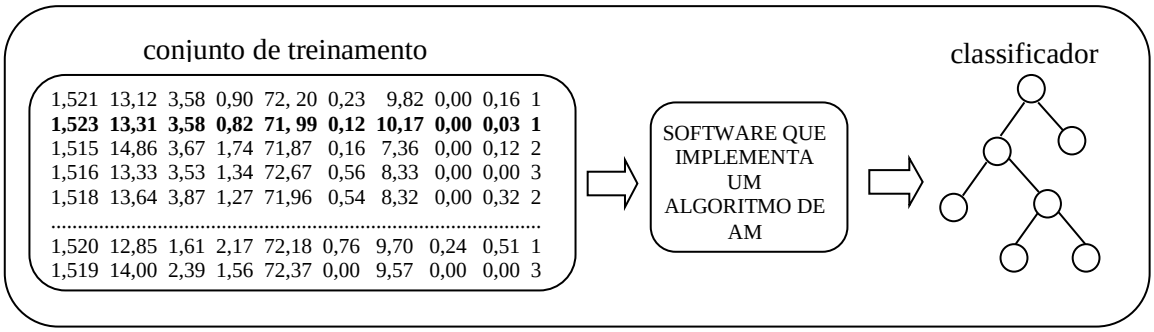


Figura 2.1 Esquema básico de funcionamento de um algoritmo supervisionado de AM. A generalização do conjunto de treinamento, feita pelo software que implementa um algoritmo de AM, é um classificador representado por uma árvore de decisão. No conjunto de treinamento estão mostradas instâncias de três classes: 1, 2 e 3.

Com o objetivo de avaliar quão representativa é a expressão do conceito induzida a partir de um conjunto de treinamento (no caso da Figura 2.1, o classificador é representado por uma árvore de decisão), um conjunto de dados, identificado como *conjunto de teste*, é utilizado.

Via de regra um *conjunto de teste* é um conjunto independente do conjunto de treinamento, cujas instâncias seguem a mesma distribuição de probabilidade daquela exibida pelo conjunto de treinamento. Um terceiro conjunto de dados, conhecido como *conjunto de validação* pode também fazer parte da metodologia de avaliação de um algoritmo de AM.

Quando se busca um modelo que seja o mais adequado para um determinado problema de aprendizado automático, usualmente a seguinte metodologia é empregada: o conjunto de treinamento é usado como entrada para diferentes algoritmos de AM (ou então, para o mesmo algoritmo, considerando diferentes valores de parâmetros e/ou diferentes configurações), de maneira que cada um deles induza um modelo; o conjunto de teste é então usado para comparar os desempenhos desses classificadores, com o objetivo de identificar o melhor `deles`. Conjuntos de validação são, então, empregados para inferir características de desempenho, tais como precisão, sensitividade e especificidade ou mesmo, o melhor modelo, etc.

A técnica conhecida como validação cruzada (*cross-validation*) é uma metodologia de validação de modelos induzidos por algoritmos de AM e consiste na repetição sistemática de processos treinamento-teste parciais [Arlot & Celisse 2010], com o objetivo de identificar o melhor modelo, dentre um conjunto de modelos induzidos. A Figura 2.2 ilustra o processo.

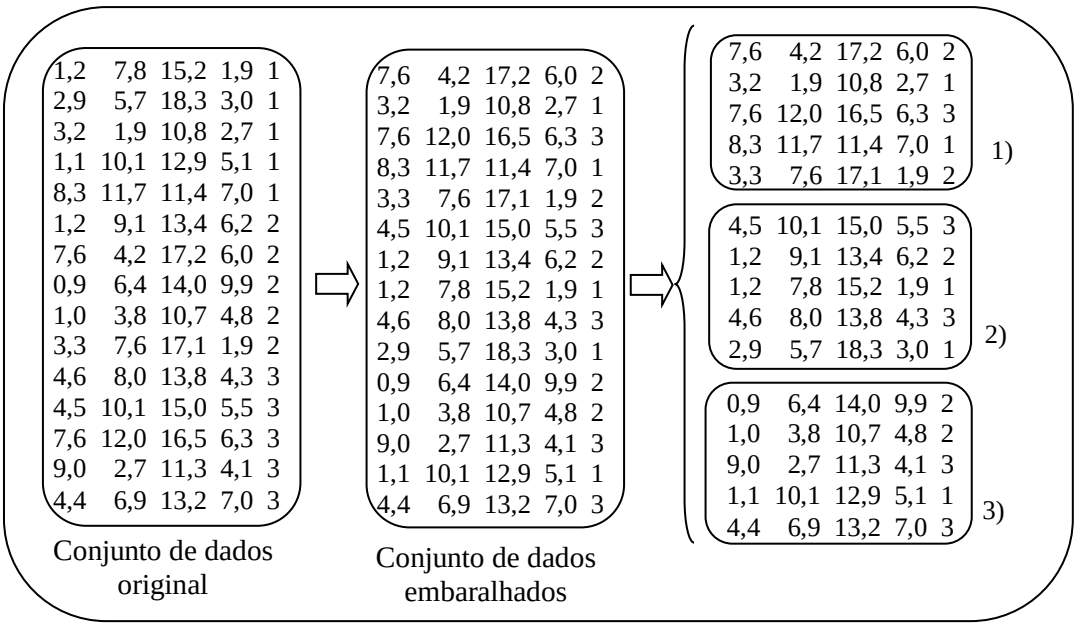


Figura 2.2 O conjunto original de dados é 'embaralhado' e, então, particionado em três subconjuntos disjuntos (1), (2) e (3) para a realização de um processo de 3-validação cruzada.

Uma k-validação cruzada consiste na divisão do conjunto de dados original em k partes aproximadamente iguais (em número de instâncias de dados) e, então, na realização de k processos de treinamento-teste, em que cada processo são usados (k−1) dos conjuntos para treinamento para a indução do classificador e o único conjunto restante, para teste do modelo induzido. A referência [Schaffer 1993] apresenta uma

revisão de várias técnicas de validação voltadas para dois problemas fundamentais em reconhecimento de padrões: seleção de modelos e estimativa de desempenho.

Considerando que muitos conjuntos de dados disponibilizados em repositórios podem ter uma certa ordenação; por exemplo, alguns agrupam dados de uma mesma classe dispondo-os em sequência, no arquivo original de dados. Quando tal arquivo é particionado em k partes, provavelmente em muitas das partes a distribuição de classes ficará desbalanceada, situação que interfere na qualidade do conceito induzido quando cada uma delas for utilizada como conjunto de treinamento.

É prática, portanto, construir cada um dos k grupos de dados selecionando randomicamente o dado do conjunto original e eliminando-o do conjunto original (de maneira a não poder ser escolhido novamente) ou, então, 'embaralhar' os dados originais de maneira a não manter a sequência em que eles são originalmente apresentados no arquivo e, só então, dividir o arquivo em k arquivos.

Em ambientes computacionais de AM em que são usados algoritmos não supervisionados, como é o caso do algoritmo foco deste trabalho, o k-Means, os conceitos de conjunto de teste e conjunto de validação, com vistas à uma avaliação do conceito induzido a partir de um conjunto de treinamento, não se aplicam devido, principalmente, à ausência da informação sobre a classe de cada instância de dado disponibilizada para o aprendizado. O fato da classe à qual a instância pertence não estar presente impede a implementação de processos de validação cruzada comumente utilizados por algoritmos supervisionados, como visto no início desta seção.

Como será visto em detalhes no Capítulo 3, devido às características inerentes ao aprendizado não supervisionado, a avaliação do conceito induzido por tais algoritmos, particularmente pelos algoritmos de agrupamento, se fundamenta basicamente em medidas estatísticas, que são utilizadas na composição das métricas de avaliação denominadas índices de validação, também abordadas no Capítulo 3.

2.4 Considerações Finais

Este capítulo teve como objetivo abordar, de maneira geral, o que é Aprendizado de máquina (AM) e suas características, assim como a especificação de uma taxonomia desses algoritmos em função da existência, ou não, de supervisão externa. Tal critério (supervisão externa) permite organizar algoritmos de AM em três grupos *i.e.*, aqueles que implementam aprendizado supervisionado, os que

implementam aprendizado não supervisionado e, finalmente, aqueles que implementam o aprendizado semissupervisionado.

Na sequência, o Capítulo 3 apresenta, de forma mais detalhada, os conceitos sobre o aprendizado não supervisionado via agrupamentos (*clustering*), que é a área específica em que esse trabalho de pesquisa se insere.

Capítulo 3

Aprendizado Não Supervisionado Utilizando Algoritmos de Agrupamentos & Algoritmo k-Means

Como informado no Capítulo 1, no *aprendizado supervisionado* cada instância de dado que participa do conjunto de treinamento é descrita por um conjunto de atributos e por uma classe associada. Geralmente a classe de cada instância de dado é determinada por um especialista humano da área de conhecimento à qual os dados se referem. A informação da classe é utilizada para guiar o processo de indução do conceito, que generaliza o conjunto de dados em expressões (representadas por regras, árvores de decisão, etc.). São várias as situações, entretanto, em que:

(1) a classe de cada instância de dado é desconhecida;

(2) não existe um especialista humano que seja capaz de, com base na descrição dos valores dos atributos das instâncias de dados, estabelecer a classe de cada uma delas ou, então,

(3) a classe é obtida apenas por meio de um processo custoso (em termos de tempo, de recursos computacionais usados, etc.) e, portanto, difícil de ser aplicado.

No contexto em que a classe à qual a instância de treinamento pertence não estar incorporada à sua descrição, métodos de AM que implementam *aprendizado não-supervisionado* devem ser considerados.

Dado um conjunto inicial de instâncias de dados X, algoritmos não-supervisionados identificam subconjuntos de instâncias de X que compartilham certas similaridades. Neste tipo de aprendizado a tarefa de um algoritmo não-supervisionado é identificar grupos de instâncias de dados, usando como critério as semelhanças ou diferenças entre eles [Kaufman & Rousseeuw 2005].

Considere inicialmente um conjunto X, contendo instâncias de dados, em que cada instância de dado é descrita por meio de um conjunto de características

(atributos). Definido de uma maneira simplista, agrupamento é um processo que identifica dissimilaridades entre as instâncias de dados e, com base no grau de dissimilaridade, as instâncias de X são particionadas em grupos. O particionamento de X deve ser tal que cada grupo agregue instâncias de dados que são mais semelhantes entre si do que semelhantes àquelas agregadas em outros grupos.

O processo de agrupar instâncias de dados com base em medidas de similaridade (ou dissimilaridade) entre elas pode ser trivialmente realizado por humanos porém, criar um algoritmo passível de ser implementado computacionalmente, de maneira a automatizar tal processo, não é uma tarefa trivial. Um algoritmo com esse propósito deverá identificar grupos de instâncias com base apenas em suas descrições.

Como apontado em [Aggarwal & Reddy 2013], técnicas de agrupamento eficientes são consideradas um grande desafio, principalmente devido ao fato de não terem supervisão externa, o que implica total desconhecimento prévio da estrutura interna do conjunto de dados (distribuição espacial, volume, densidade, formas geométricas dos grupos, etc.). Nesse contexto o aprendizado automático passa a ser uma atividade exploratória, para verificar quais são os grupos de dados estatisticamente separáveis (ou não), quais os grupos mais evidentes e sua relação com o que se deseja discriminar, em uma tentativa de evidenciar a estrutura subjacente ao conjunto de dados, tendo como informação suas descrições, cada uma delas representada por um vetor de atributos.

Apesar de não ser o foco deste trabalho, é importante lembrar que existem muitas outras peculiaridades em relação ao conjunto de instância de dados disponibilizado, tais como

(1) descrição de cada instância envolvendo vários tipos de atributo (numérico, ordinal, simbólico, estruturado, etc.;
(2) instâncias de dados com valores ausentes de determinados atributos;
(3) instâncias repetidas;
(4) ruídos nos valores de determinados atributos;
(5) número insuficiente de atributos na caracterização da instância;
(6) presença de *outliers*, e muitos outros.

É importante ressaltar, também, que cada uma dessas particularidades se apresenta como um problema a ser tratado antes do aprendizado automático (seja

ele supervisionado ou não-supervisionado) acontecer, caso contrário vão interferir na obtenção de uma expressão representativa do conceito subjacente aos dados.

3.1 O Processo de Agrupamento de Dados

Devido tanto à complexidade quanto ao caráter exploratório de um processo de agrupamento, diversos pesquisadores têm publicado propostas de roteiro para organizar as atividades envolvidas no processo. Jain e colaboradores, como descrito em [Jain & Dubes 1988] e [Jain 2010], propõem e descrevem cinco passos que devem ser observados para a utilização de alguma técnica de agrupamento. Theodoridis e Koutroumbas em [Theodoridis & Koutroumbas 2009] também sugerem um procedimento semelhante ao proposto por Jain e colaboradores. A sequência de passos envolvidos em processos de análise de dados que fazem uso de algoritmos de agrupamento pode ser descrita, de uma maneira geral, como sugerido em [Theodoridis & Koutroumbas 2009], pelos passos:

Passo 1 – representar as instâncias de dados a serem agrupadas, por meio da escolha de um grupo de atributos que efetivamente sejam relevantes para caracterizar as instâncias de dados disponibilizadas, com vistas à serem fornecidas a um processo de agrupamento;

Passo 2 – selecionar a medida de similaridade ou dissimilaridade mais adequada ao domínio de conhecimento em questão, e escolher qual o critério de agrupamento, conforme tendência verificada intuitivamente. Esses critérios, em geral, são medidas de distância entre pares de dados do conjunto e são definidos conforme o conhecimento/experiência do especialista humano. Existem inúmeras funções matemáticas que podem ser utilizadas. A escolha depende do tipo de representação dos atributos ou da tendência do agrupamento; uma das medidas mais simples e populares utilizada é a distância euclidiana.

Passo 3 – escolher e utilizar um ou mais algoritmos de agrupamento na construção dos grupos, com a escolha do(s) algoritmo(s) mais adequado(s) e de acordo com critérios e medidas escolhidas. Algoritmos podem produzir resultados diferentes uns dos outros; isso se deve ao fato das estratégias utilizadas não serem as mesmas.

Passo 4 – obter uma "abstração dos dados", ou seja, uma representação simples e compacta do conjunto de dados, com vistas tanto à interpretabilidade humana quanto à

simplicidade computacional, com o objetivo de permitir um processamento posterior eficiente.

Passo 5 – interpretar, avaliar e validar os resultados do processo de agrupamento. É comum a necessidade de comparar os resultados de um algoritmo de agrupamento a evidências e análises experimentais, com objetivo de assegurar a robustez do processo de inferência de conclusões. O(s) grupo(s) gerado(s) por um algoritmo de agrupamento precisam, então, ser validados, para ratificar, ou não, sua corretude. Geralmente processos de validação utilizam medidas estatísticas tais como taxas de erros ou índices de correlação que permitem estimar a validade do(s) resultados obtidos.

3.2 Notação e Definições Básicas

Dado o volume considerável de material bibliográfico associado à área de pesquisa de agrupamentos e, particularmente, às muitas diferentes notações empregadas, essa seção tem por objetivo adequar e padronizar a notação utilizada neste trabalho bem como apresentar algumas das definições utilizadas ao longo do trabalho.

Em muitas publicações sobre agrupamento de dados, diferentes termos são utilizados para expressarem o mesmo conceito. É o caso, por exemplo, das expressões: ponto de dados, padrão de dados, casos, observações, objetos, indivíduos, itens e tuplas, que são usadas para nomear um elemento do conjunto de dados *i.e.*, uma instância de dados usada pelos algoritmos de agrupamento. Também, com relação à linguagem utilizada para a representação de um dado, é comum encontrar: termos, variáveis, atributos, medidas ou características.

De maneira geral um conjunto com N instâncias de dados é denotado como $CD = \{D_1, D_2, ..., D_N\}$. A notação para representar cada uma das instâncias, descritas por M atributos *i.e.*, $A_1, A_2, ..., A_M$, é como o vetor $(D_{i_1}, D_{i_2}, . . . D_{i_M})$, $\forall$ i =1, ..., N, em que D_{i_j} é um dos possíveis valores do atributo A_j, j = 1, ..., M. O número de atributos utilizados para descrever o conjunto de dados *i.e.*, M, é referenciado como a dimensão do dado. Optou-se, entretanto, nas inúmeras descrições de algoritmos investigados e apresentados no texto, por manter a notação utilizada pelo(s) autor(res) dos respectivos algoritmos, de maneira a reproduzir o(s) algoritmo(s) como foi(ram) publicado(s) na(s) referência(s) citada(s) evitando, com isso, a possibilidade de incorreções ao mudar notações.

Atributos podem assumir valores quantitativos em intervalos contínuos ou em um conjunto finito discreto, como por exemplo, no conjunto {0,1} ou, então, valores qualitativos (categóricos: nominal ou ordinal). Como apontam vários trabalhos, a caracterização do tipo de atributo direciona o processo de seleção da medida de similaridade que será utilizada [Gowda & Diday 1992], [Kaufman & Rousseeuw 2005].

Considerando o conjunto de dados CD = $\{D_1, D_2, ..., D_N\}$, e um número inteiro k, um k-agrupamento de CD é definido com uma partição de CD em k grupos (subconjuntos), $G_1, G_2, ..., G_k$ ou seja, k-agrupamento = $\{G_1, G_2, ..., G_k\}$. De acordo com a definição matemática de partição de um conjunto, as seguintes condições devem, pois, ser satisfeitas:

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

(2) A união de todos os grupos recompõe o conjunto inicial, ou seja:

$CD = \bigcup_{i=1}^{k} G_i$

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

Assume-se que os dados agrupados em G_i (i = 1, ..., k) são "mais semelhantes" entre si do que a dados que pertencem a outros grupos. Na maioria das vezes um conjunto com N dados M-dimensionais é tratado por algoritmos como uma matriz $N \times M$.

3.3 Medidas de Dissimilaridade (Distância)

Como comentado anteriormente, o objetivo de um algoritmo de agrupamento é, a partir de um conjunto inicial de instâncias de dados, definir um conjunto de grupos de instâncias de tal maneira que exista o máximo de homogeneidade entre as instâncias que pertencem a cada um dos grupos e o máximo de heterogeneidade entre instâncias que pertencem a grupos diferentes. A dissimilaridade é, via de regra, implementada por meio da distância euclidiana.

Um método de agrupamento considerado bom é aquele que induz grupos de qualidade com alta similaridade intragrupos e baixa similaridade intergrupos. Essa qualidade dos resultados depende tanto da medida de dissimilaridade usada pelo método, quanto da maneira como é implementada. Outras medidas são também utilizadas, na dependência do domínio de dados em questão, tais como a distância

euclidiana média, a de Hellinger, a variacional e a de Mahalanobis e Hamming (ver [Anderberg 1973] para detalhes).

O método de agrupamento de interesse deste trabalho (*i.e.*, k-Means) tem um grande número de aplicações em que dados são descritos por atributos com valores contínuos e, geralmente, a distância euclidiana é utilizada, principalmente quando o conjunto de dados tem grupos 'compactos' ou 'bem separados' como apontado em [Jain 2010].

3.4 O Algoritmo k-Means

Foi de interesse no trabalho de pesquisa desenvolvido um dos mais conhecidos algoritmos de agrupamento, aquele conhecido como k-Means [MacQueen 1967] que, desde a sua proposta em 1967, continua sendo usado em uma grande diversidade de domínios de dados, devido à sua simplicidade, fácil implementação e rapidez em execução.

3.4.1 Considerações Iniciais

O k-Means é caracterizado como um algoritmo particional que, dado um conjunto de instâncias de dados como entrada, tem por objetivo encontrar uma partição do conjunto em k grupos disjuntos. Como já especificado no próprio nome do algoritmo, o valor de k é também entrada para o algoritmo, fornecido pelo usuário, e que representa o número de grupos que o agrupamento, a ser induzido pelo algoritmo, deve ter. O k-Means inicia a construção do agrupamento por meio da escolha randômica dos centroides dos k grupos (de instâncias de dados) a serem construídos.

Para um dado conjunto de instâncias de dados e, devido ao fato da escolha inicial dos k centroides de grupos ser feita de maneira randômica pelo algoritmo, o k-Means nem sempre induz o mesmo agrupamento, em duas execuções distintas do mesmo algoritmo, com o mesmo conjunto de dados e o mesmo valor para o parâmetro k. Esse fato pode ser um problema em certos domínios de dados. Na literatura podem ser encontrados vários métodos que buscam resolver o problema da inicialização dos centroides de grupos, bem como alguns trabalhos que fazem revisões de alguns desses métodos, tais como as revisões apresentadas em [Peña *et al.* 1999] [Khan & Ahmad 2004] [Celebi *et al.* 2013] [Celebi 2015].

3.4.2 Detalhes do k-Means

Como resumidamente apresentado em [Witten *et al.* 2011], tendo como entrada um conjunto, CD = {D_1, D_2, ..., D_N}, contendo N instâncias de dados (ou pontos), e um valor (inteiro) atribuído ao parâmetro k, o algoritmo k-Means inicia escolhendo, randomicamente, k instâncias de CD, que representam k centroides de grupos (centroide é caracterizado como a média dos valores dos atributos entre as instâncias associados a um grupo).

Cada instância de CD, então, é atribuída ao grupo cujo centroide lhe seja mais próximo, por meio do cálculo da distância (euclidiana, geralmente) de cada instância, a cada um dos k centroides de grupos considerados. A seguir, a média dos valores dos atributos entre as instâncias atribuídas a cada grupo (a média dos valores dos atributos entre as instâncias representa o centroide do respectivo grupo) é calculada. Esses centroides passam, então, a ser os novos centroides de grupos e todo o processo é repetido, com os novos centroides de grupos.

O processo iterativo continua até que as mesmas instâncias sejam atribuídas aos mesmos grupos, em iterações consecutivas, um indicativo que os centroides de grupos atingiram estabilidade e assim permanecerão. Uma vez que o processo iterativo tenha se estabilizado, cada instância é atribuída ao grupo associado ao seu centroide de grupo mais próximo, processo que pode ser matematicamente parafraseado como tendo efeito de minimizar o total dos quadrados das distâncias de todas as instâncias aos seus respectivos centroides de grupos. Esse mínimo, entretanto, é local e não existe garantia que seja um mínimo global.

Os grupos resultantes de um agrupamento induzido pelo k-Means são tão sensíveis à escolha inicial dos centroides de grupos que uma pequena mudança no conjunto dos centroides de grupos escolhidos inicialmente, pode implicar a criação de um agrupamento completamente diferente de um obtido anteriormente. Para a obtenção de bons resultados com o k-Means, usualmente o que se faz na prática é executá-lo um determinado número de vezes e, a cada vez, com um conjunto diferente de centroides de grupos até obter bons grupos resultantes.

Como apontado em [Han *et al.* 2012], a complexidade em tempo do k-Means é dada por O(Nkt), em que N é o número total de padrões, k é o número de grupos e t é o número de iterações. Via de regra tem-se $k << N$ e $t << N$, o que torna o algoritmo relativamente escalável e eficiente, quando do processamento de um grande volume de

dados. Algoritmo 3.1 apresenta um pseudocódigo simplificado do algoritmo k-Means, inspirado naquele encontrado em [Han *et al.* 2012] e [Witten *et al.* 2011], usando a notação encontrada em [Witten *et al.* 2011].

Na literatura podem ser encontradas diversas variações do k-Means original e, geralmente, tais variações diferem em relação à seleção inicial dos centroides de grupos, cálculo da dissimilaridade (distância) e estratégias para o cálculo dos centroides. Como já informado anteriormente, a investigação conduzida e apresentada nesta dissertação teve por foco o processo da seleção inicial dos centroides de grupos.

procedure *k-means(CD,k,AG)*
input: CD = $\{D_1, D_2, ..., D_N\}$ % conjunto de instâncias de dados a serem agrupadas.
k % número de grupos a ser criado.
output: AG = $\{G_1,G_2,...G_k\}$ % agrupamento formado por k grupos de instâncias de dados.
begin
(1) escolher arbitrariamente k instâncias $\in$ CD, cada um como centroide dos grupos $G_1,G_2,...G_k$
% após (1) cada um dos k grupos contém apenas o centroide
(2) **repeat**
(3) (re)atribuir cada instância $D_i \in$ CD ao grupo associado ao centroide que lhe seja mais próximo;
(4) atualizar os centroides de cada um dos k grupos, como a média dos valores dos atributos entre as instâncias a ele associados;
(5) **until** nenhuma alteração aconteça no agrupamento.
end
return AG = $\{G_1,G_2,...G_k\}$
end procedure

Algoritmo 3.1 Pseudocódigo do algoritmo k-Means.

3.5 Índices de Validação

A validação (*i.e.*, o quão representativo é o agrupamento induzido por um algoritmo de agrupamento) é reconhecidamente um processo essencial para a confiabilidade e o uso de algoritmos de agrupamento em ambientes automatizados utilizados em qualquer área do conhecimento.

Como apontado em [Maulik & Bandyopadhyay 2002], duas perguntas precisam ser endereçadas por qualquer sistema computacional que implementa algoritmos de agrupamento: (1) quantos grupos estão realmente presentes no conjunto de instâncias de dados considerado? e (2) quão real (e representativo) é o agrupamento induzido? Independentemente do algoritmo de agrupamento utilizado, é sempre preciso determinar o número de grupos do agrupamento e avaliar o quão representativo é o conjunto de grupos (*i.e.*, o agrupamento) para refletir a organização dos dados fornecidos.

Índices (ou medidas de validação) podem ser abordados como pertencentes a duas categorias: validação externa e validação interna. A principal diferença entre essas duas

categorias está no uso (ou não) de informação externa que, tipicamente, envolve o uso do atributo classe, previamente determinado por uma fonte externa (usuário, por exemplo). Em muitas situações práticas, entretanto, informações externas tais como as classes de cada instância de dado, não estão disponíveis.

A parte de experimentação envolvendo os vários algoritmos de inicialização do k-Means, apresentada em detalhes no Capítulo 6, usou tanto um índice externo, especificamente, o índice Rand [Rand 1971], usando a informação relativa à classe associada a cada instância e três índices internos baseados em medidas estatísticas, o de Dunn [Dunn 1974], o Silhouette [Rousseeuw 1987] e o Davies-Bouldin [Davies & Bouldin 1979]. Uma breve descrição de cada um dos quatro índices, iniciando com aquela do índice externo, é apresentada na sequência.

3.5.1 Índice Rand

O valor do índice de validação conhecido como índice Rand [Rand 1971] pode ser abordado como uma medida de similaridade entre dois agrupamentos. Nos experimentos apresentados no Capítulo 6 desta dissertação, um dos agrupamentos será aquele induzido pelo k-Means (usando um dos quatro algoritmos de inicialização) e o outro agrupamento, será aquele providenciado por meio da informação externa dada pelas classes associadas às instâncias de dados. O índice Rand pode ser abordado formalmente da seguinte maneira descrita a seguir.

Considere X o conjunto de instâncias de dados a serem agrupadas. Considere que um dos agrupamentos de X é notado por $Y = \{Y_1, Y_2, ..., Y_{NY}\}$ e o outro, por $Z = \{Z_1, Z_2, ..., Z_{NZ}\}$. Para a determinação do índice Rand associado aos dois agrupamentos, é necessário inicialmente o cálculo dos valores a, b, c e d, como mostrados a seguir e, com base neles, determinar o valor do índice Rand associado, como descreve a Eq. (1).

- a: número de pares de instâncias de dados de X que estão em um mesmo grupo no agrupamento Y e no mesmo grupo no agrupamento Z;
- b: número de pares de instâncias de dados de X que estão em grupos diferentes no agrupamento Y e em grupos diferentes no agrupamento Z;
- c: número de pares de instâncias de dados de X que estão em um mesmo grupo no agrupamento Y e em grupos diferentes no agrupamento Z e

- d: número de pares de instâncias de dados de X que estão em grupos diferentes no agrupamento Y e no mesmo grupo no agrupamento Z.

$$R = (a+b)/(a+b+c+d) \quad (1)$$

3.5.2 Índice Dunn

Considere um conjunto X contendo N instâncias de dados, que foram agrupadas em k grupos, G_1, G_2, ..., G_k, formando o agrupamento AG = {G_1, G_2, ..., G_k}. Seja G_i o i-ésimo grupo de AG, i = 1, ..., k, contendo n_i instâncias de dados. Considere x e y duas instâncias de dados de X, que distam uma da outra de dist(x,y). O índice de Dunn (D) associado a AG pode ser formalmente definido pela Eq. (1).

$$D = \min_i\{\min_j(A/B)\} \quad (1)$$

em que

$$A = \min_{x \in C_i, y \in C_j} \text{dist}(x,y) \text{ and } B = \max_k\{\max_{x,y \in C_k} \text{dist}(x,y)\}.$$

O índice de Dunn leva em conta as características de densidade e de separação de grupos e é tal que $D \in [0, \infty)$.

O índice de Dunn pode também ser definido pela Eq. (2), em que dist_{min} é a menor distância entre duas instâncias de dados pertencentes a grupos diferentes e dist_{max} é a maior distância entre duas instâncias de dados em um mesmo grupo.

$$D = \text{dist}_{min}/\text{dist}_{max} \quad (2)$$

Grupos compactos têm valores maiores de dist_{min} e menores valores de dist_{max} e, portanto, quanto maior for o valor de D, melhor é o agrupamento induzido AG.

3.5.3 Índice Silhouette

Considere um conjunto X contendo N instâncias de dados *i.e.*, X = {X_1, X_2, ..., X_N}, que foram agrupadas em k grupos, G_1, G_2, ..., G_k, formando o agrupamento AG = {G_1, G_2, ..., G_k}.

O índice de validação de agrupamentos conhecido como Silhouette, proposto em [Rousseeuw 1987], permite avaliar quão ajustada cada instância de dados está, ao grupo de AG ao qual pertence. O índice avalia cada instância em relação à sua similaridade às outras instâncias do grupo ao qual pertence, comparado os valores com aqueles de instâncias pertencentes a outros grupos.

Seja G_i o i-ésimo grupo de AG, i = 1, ..., k, contendo n_i instâncias de dados. Considere X_i e X_j duas instâncias de dados de X, que distam uma da outra de dist(X_i, X_j). O índice Silhouette (S) pode ser formalmente definido pela Eq. (3), em que |AG| = k.

$$S = \frac{1}{|AG|} \sum_i \left\{ \frac{1}{|G_i|} \sum_{X_i \in G_i} \frac{b(X_i) - a(X_i)}{\max[b(X_i), a(X_i)]} \right\} \tag{3}$$

em que a função $a(X_i)$ está definida na Eq. 4 e a função $b(X_i)$ na Eq. 5.

$$a(X_i) = \frac{1}{|G_i| - 1} \sum_{X_j \in G_i} \mathrm{dist}(X_i, X_j) \tag{4}$$

$$b(X_i) = \min_{j, j \neq i} \left[\frac{1}{|G_j| - 1} \sum_{X_j \in G_j} \mathrm{dist}(X_i, X_j) \right] \tag{5}$$

O valor do índice Silhouette associado a um determinado agrupamento G varia no intervalo de –1 a +1 *i.e.*, S $\in$ [–1 +1] e pode ser calculado com qualquer métrica de distância. Neste trabalho foi utilizada a distância euclidiana. Quanto mais próximo de 1 for o valor do índice S associado a um determinado agrupamento obtido, melhor é o agrupamento.

3.5.4 Índice Davies & Bouldin

O índice de validação Davies & Bouldin, proposto em [Davies & Bouldin 1979], é baseado em cálculo de similaridade intragrupo e de diferenças intergrupos. É um índice que promove agrupamentos com grupos compactos e distantes entre si.

Considere novamente um conjunto X contendo N instâncias de dados *i.e.*, X = {X_1, X_2, ..., X_N}, que foram agrupadas em k grupos, G_1, G_2, ..., G_k, formando o agrupamento AG = {G_1, G_2, ..., G_k}. Seja G_i o i-ésimo grupo de AG, i = 1, ..., k, contendo n_i instâncias de dados. Considere ainda que os pontos representativos dos k grupos *i.e.*, seus respectivos centroides, sejam C = {C_1, C_2, ..., C_k}. O índice DB pode ser formalmente descrito pela fórmula Eq.(6).

$$\mathrm{DB} = \frac{1}{\mathrm{k}} \sum_{\mathrm{i}=1}^{\mathrm{k}} max(d_{ij})_{j=1\ldots k; j\neq i} \; d_{\mathrm{ij}} = \left(\frac{\delta_{\mathrm{i}} + \delta_{\mathrm{j}}}{d(\mathrm{C_i}, \mathrm{C_j})} \right) \tag{6}$$

em que δ_i, i = 1, ..., k, é a distância média entre instâncias do grupo G_i e o respectivo centroide C_i do grupo G_i, o valor δ_j, j = 1, ..., k, é a distância média entre instâncias do grupo G_j e respectivo centroide C_j do grupo G_j, e $d(\mathrm{C_i}, \mathrm{C_j})$ é a distância entre os centroides (C_i e C_j, $i \neq j$ e i,j = 1, ..., k).

O índice DB mede a similaridade média e a dispersão dos grupos. Valores baixos do índice DB correspondem a grupos que são compactos e distantes um do outro.

Na Tabela 3.1 é mostrado quando os índices são bons ou ruins, com objetivo de identificar o melhor deles foi utilizado o símbolo ↑ (seta para cima) quando o valor do índice for melhor com maiores valores e ↓ (seta para baixo) quando o valor do índice for melhor com menores valores.

Tabela 3.1 Identificação dos resultados dos índices

Índice	Resultados Bons	Resultados Ruins
Rand (↑)	Maiores valores	Menores valores
Dunn (↑)	Maiores valores	Menores valores
Silhouette (↑)	Maiores valores	Menores Valores
Davies & Bouldin (↓)	Menores valores	Maiores Valores

Capítulo 4

Algoritmos de Inicialização do K-Means: k-Means++ & SPSS

Na literatura podem ser encontrados inúmeros métodos que se propõem a sanar a deficiência do k-Means, com relação à sua proposta original de, no seu primeiro passo, selecionar randomicamente k instâncias do conjunto de dados e promove-las a centroides de k grupos. Em inúmeros trabalhos foi observado que tanto a convergência do processo iterativo quanto o desempenho do agrupamento obtido pelo k-Means dependem do conjunto inicial de centroides [Burks *et al.* 2015]. Tanto o número de grupos (k) quanto a inicialização dos centroides desses grupos são aspectos relevantes que afetam o desempenho do algoritmo.

Neste capítulo são abordadas duas propostas de inicialização do conjunto inicial de centroides de grupos encontradas na literatura, que têm como objetivo contribuir para o desempenho do k-Means. Na Seção 4.1 é abordado o algoritmo k-Means++ [Arthur & Vassilvitskii 2007] e, na Seção 4.2, é abordado o algoritmo *Single Pass Seed Selection* (SPSS) [Pavan *et al.* 2010], que é apresentado pelos autores como uma melhoria ao processo de inicialização proposto pelo k-Means++.

Como já antecipado no Capítulo 1, a única descrição do SPSS, que foi encontrada na referência [Pavan *et al.* 2010], não apresenta todos os detalhes necessários que viabilizariam sua implementação e avaliações comparativas com os resultados apresentados na referência em questão.

Os contatos via e-mail com o primeiro autor do artigo, com o objetivo de obter uma descrição com todos os detalhes necessários para subsidiar uma implementação fiel, não foram bem sucedidos. Estes detalhes deveriam ser inferidos para poder realizar a implementação do algoritmo, correndo o risco de alterar a proposta original dos autores. Entretanto, como foi feito um estudo detalhado do algoritmo, decidiu-se implementá-lo e apresentá-lo nessa dissertação, as ambiguidades foram resolvidas durante a implementação do algoritmo.

4.1 Inicialização de Centroides do Algoritmo k-Means++

A variante do k-Means, chamada k-Means++ [Arthur & Vassilvitskii 2007], se constitui no próprio k-Means, com uma alteração em seu passo de inicialização. De acordo com os autores, o k-Means++ é competitivo, fazendo k-Means convergir com complexidade O(log k) calculada após a inicialização de centroide.

O processo de inicialização do k-Means++ ainda escolhe randomicamente os k centroides iniciais, mas os pondera de acordo com o quadrado de suas distâncias àquele centroide que lhes seja mais próximo, dentre os já escolhidos. Os autores mostram empiricamente que o algoritmo k-Means++ tem melhor desempenho que o k-Means tanto em acurácia quanto em velocidade e, geralmente, por uma margem substancial.

Algoritmo 4.1 exibe o pseudocódigo do k-Means++, como proposto em [Arthur & Vassilvitskii 2007], em que X é o conjunto de pontos a ser agrupado. O algoritmo faz uso de uma função $D: X \rightarrow \Re$ (conjunto dos números reais), em que D(x) representa a menor distância de uma instância $x \in X$, ao centroide que lhe é mais próximo, dentre os centroides já escolhidos.

Uma vez feita a inicialização dos k centroides, o k-Means++ repete o próprio procedimento do k-Means que, em Algoritmo 4.1, é representado pela chamada ao procedimento *k-Means_without_initialization*(X,{c_i, c_2, ..., c_k},k,AG), detalhado no Algoritmo 4.2. Vale notar que o Algoritmo 4.2 é o Algoritmo 3.1 ligeiramente modificado, em que a parte de inicialização foi eliminada, uma vez que os k centroides iniciais são passados ao algoritmo, como parâmetro.

procedure *k-Means++(X,k,AG)*
input: X % conjunto de N instâncias.
k % número de grupos.
output: AG = {G_1, G_2, ...,G_k}

begin
% processo de inicialização dos centroides (i)
(i_0) C ← {}
(i_1) eleja randomicamente um centroide, $c_1 \in X$ (C ← C ∪ {c_1})
(i_2) pegue um novo centroide c_i, escolhendo $x \in X$ com maior probabilidade $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$, então faça C ← C ∪ {$c_i$}.
(i_3) repita o passo (i_2) até que tenham sido pegos k centroides.
% finalização do processo de inicialização dos centroides
call *k-Means_without_initialization*(X,{c_i, c_2, ..., c_k},k,AG) % Algoritmo 4.2.
end
return AG = {G_1, G_2, ...,G_k}
end procedure

Algoritmo 4.1 Pseudocódigo do k-Means++, uma variante do k-Means com um processo próprio de inicialização.

k-Means_without_initialization(X,{c_i, c_2, ..., c_k},k,AG)
input: X % conjunto com N instâncias.
k % número de grupos.
{c_i, c_2, ..., c_k} % k centroides criados pelo processo de inicialização do k-Means++.
output: AG = {G_1, G_2, ...,G_k}
begin
1. para cada i $\in$ {1, 2, 3, ..., k} estabeleça o grupo G_i como sendo constituído por aquelas instâncias de X, que estão mais perto de c_i do que de qualquer outro c_j, para todo j $\neq$ i.
2. para cada i $\in$ {1, 2, 3, ..., k}, faça c_i ser o ponto médio de todas as instâncias em G_i, ou seja, $c_i = 1/|G_i| \times (\Sigma_{x \in G_i} x)$
3. repita passos 1. e 2. até que C não mude mais.

end
return AG = {G_1, G_2, ...,G_k}
end procedure

Algoritmo 4.2 Pseudocódigo do k-Means (sem o passo de inicialização), para ser usado no k-Means++ (Algoritmo 4.1).

O processo de inicialização do k-Means++ não escolhe como o próximo centroide uma instância que esteja mais distante dos centroides já escolhidos mas sim, escolhe uma instância com uma probabilidade proporcional à sua distância aos centroides já escolhidos.

Como apontado em [Pavan *et al.* 2010], devido à seleção randômica do primeiro centroide e às seleções probabilísticas dos centroides restantes, diferentes execuções devem ser realizadas para a obtenção de um bom agrupamento. Particularmente, o passo (i_2) do Algoritmo 4.1 pode ser implementado de diversas maneiras.

(i_2) pegue um novo centroide c_i, escolhendo $x \in X$ com maior probabilidade $D(x)^2/(\Sigma_{x \in X} D(x)^2)$ ($C \leftarrow C \cup \{c_i\}$)

A distribuição de probabilidade descrita em (i_2) geralmente é implementada pelo método *seleção proporcional à aptidão,* o cálculo das probabilidades acumulativas de cada uma das N instâncias de X é utilizado. Essas probabilidades acumulativas são subintervalos do intervalo [0 1], para escolha dos centroides uma seleção aleatória é necessária. Esta seleção escolhe um valor real no intervalo [0,1], esse valor é comparado entre cada elemento do vetor de probabilidades acumuladas até que se encontre a instância com maior valor de probabilidade associado e escolhe como próximo centroide.

O pseudocódigo apresentado em Algoritmo 4.3 foi escrito com o intuito único de promover o entendimento do processo, sem qualquer preocupação com a sua eficiência;

via de regra tal procedimento pode ter sua eficiência promovida por meio do uso de estruturas de dados convenientes (tal como a de k-d tree [Bentley 1975]).

```
procedure inicialization(X,k)
input:  X              % conjunto com N instâncias.
        k              % número de grupos.
        C              % vetor de centroides com dimensão k, com apenas o primeiro elemento.
output: vetor D = [d1, d2, ..., dN]   % vetor de distâncias.
        vetor P = [p1, p2, ..., pN]   % vetor de probabilidades – valores proporcionais a D(x)^2.
        vetor C = [c1, c2, ..., ck]   % vetor de centroides.
begin
  1. C[1] ← random_choice(X)
  2. j ← 2
  3. while j ≤ k do
  4.    begin
  5.       soma ← 0
  6.       for i ← 1 to N do
  7.       begin
  8.          d[i] ← distance_to_nearer_centroid(xi,C)        %distância ao quadrado.
  9.          soma ← soma + d[i]
 10.       end
 11.       for i ← 1 to N do p[i] ← d[i]/soma
 12.       c[j] ← element_with_highest_probability(P)
.13        j ← j + 1
 14.   end
 15. return D, P, C
end procedure
```

Algoritmo 4.3 Pseudocodigo do procedimento de inicialização do k-Means++.

4.1.1 Primeiro Exemplo do Processo de Inicialização do k-Means++ (Espaço Unidimensional)

Considere um conjunto X contendo seis instâncias, X = {x_1, x_2, x_3, x_4, x_5, x_6} = {1, 2, 3, 4, 5, 6}, extraídas do espaço unidimensional, como mostra a Figura 4.1 e suponha que a tarefa de agrupamento seja a de agrupar as seis instâncias em k=3 grupos.

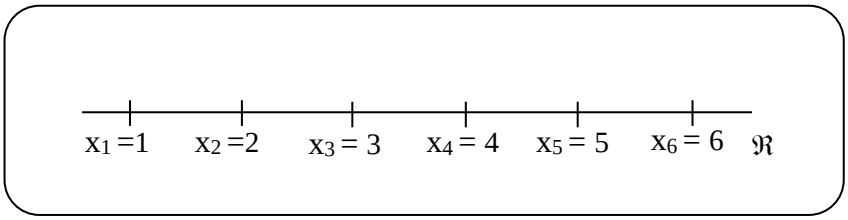


Figura 4.1 Conjunto X de instâncias de dados, para ser agrupado em k=3 grupos. Foco na inicialização do k-Means++.

Considere que a chamada ao procedimento *random_choice*(X) tenha como resultado x_1. Portanto, o vetor de centroides terá a configuração mostrada como segue.

C[1]	C[2]	C[3]
x_1 = 1	–	–

Note que, até o momento, apenas um centroide foi escolhido. Considerando agora o passo 8 do Algoritmo 4.3,

8. d[i] ← *distance_to_nearer_centroid*(x_i,C) % distância ao quadrado

o procedimento *distance_to_nearer_centroid* vai calcular, para cada instância de X, a sua distância (ao quadrado) ao centroide que lhe seja mais próximo, construindo assim um vetor de distâncias ao quadrado, de dimensão igual a N = 6 (número de instâncias de X). Como existe até o momento apenas 1 centroide, c_1, serão calculadas as distâncias ao quadrado de cada uma das instâncias ao c_1 = C[1] = 1, resultando no vetor de distâncias ao quadrado (d), mostrado a seguir.

d[1] = $d^2(x_1,c_1)$	d[2] = $d^2(x_2,c_1)$	d[3] = $d^2(x_3,c_1)$	d[4] = $d^2(x_4,c_1)$	d[5] = $d^2(x_5,c_1)$	d[6] = $d^2(x_6,c_1)$
0	1	4	9	16	25

Somando todos os N (= 6) elementos do vetor d resulta em soma = 0 + 1 + 4 + 9 + 16 + 25 = 55 e, portanto, após o passo 11 do Algoritmo 4.3, dado por:

11. **for** i ← 1 to N **do** p[i] ← d[i]/soma

tem-se que o vetor de probabilidades, p, associado ao vetor de distâncias, é:

p[1] = d[1]/55	p[2] = d[2]/55	p[3] = d[3]/55	p[4] = d[4]/55	p[5] = d[5]/55	p[6] = d[6]/55
0/55 = 0,00000	1/55 = 0,01818	4/55 = 0,07272	9/55 = 0,16363	16/55 = 0,29090	25/55 = **0,45454**

O chamado *vetor de probabilidade acumulada*, pa, associado ao vetor p é dado por:

pa[1] = p[1]	pa[2] = pa[1]+p[2]	pa[3]=pa[2]+p[3]	pa[4]=pa[3]+p[4]	pa[5]=pa[4]+p[5]	pa[6]=pa[5]+p[6]
0,00000	0,01818	0,09090	0,25453	0,54543	0,99997≅ 1,0

Considerando o vetor pa, a escolha do próximo centroide novamente utiliza a seleção proporcional, um sorteio entre [0,1] é realizado, suponha que o valor sorteado seja 0,67, esse valor é comparado com cada elemento do vetor de probabilidade acumulada até encontrar qual a primeira instância tem maior valor de probabilidade associado, na última comparação, x_6 é a instancia associada ao vetor com maior probabilidade acumulada, portando, x_6 é escolhida como o segundo centroide. O conjunto X, com a escolha do segundo centroide, está mostrado na Figura 4.2 e o vetor de centroides na sequência.

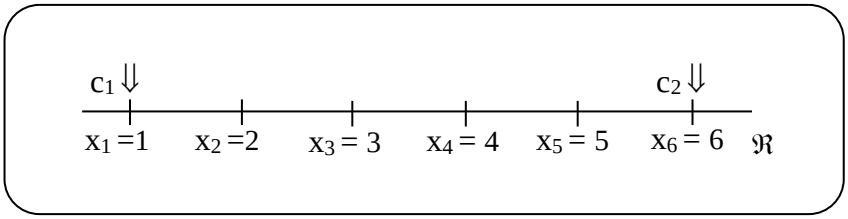


Figura 4.2 Conjunto X de instâncias de dados, após a inicialização de dois centroides, c_1 (escolhido randomicamente) e c_2, cuja escolha foi definida com base em distâncias. Ambos estão assinalados com flechas na figura.

C[1]	C[2]	C[3]
1 (x_1)	6 (x_6)	–

Como foram escolhidos apenas 2 centroides, o processo volta a se repetir (*loop* do **while** – linhas 3 a 14 no Algoritmo 4.3), para a escolha do terceiro centroide. O cálculo do vetor d, agora, considera os dois centroides já escolhidos. O vetor contendo os valores das distâncias ao quadrado, de cada instância de X ao centroide que lhe é mais próximo é:

$d[1] = d^2(x_1,c_1)$	$d[2] = d^2(x_2,c_1)$	$d[3] = d^2(x_3,c_1)$	$d[4] = d^2(x_4,c_2)$	$d[5] = d^2(x_5,c_2)$	$d[6] = d^2(x_6,c_2)$
0	1	4	4	1	0

Somando todos os N (= 6) elementos do vetor d resulta em soma = 0 + 1 + 4 + 4 + 1 + 0 = 10 e, portanto, após o passo 11 do Algoritmo 4.3, dado por:

11. **for** i ← 1 to N **do** p[i] ← d[i]/soma

tem-se que o vetor de probabilidades, p, associado ao vetor de distâncias, é:

p[1] = d[1]/10	p[2] = d[2]/10	p[3] = d[3]/10	p[4] = d[4]/10	p[5] = d[5]/10	p[6] = d[6]/10
0/10 = 0,00000	1/10 = 0,10000	4/10 = 0,40000	4/10 = 0,40000	1/10 = 0,10000	0/10 = 0,00000

O chamado *vetor de probabilidade acumulada*, pa, associado ao vetor p é dado por:

pa[1] = p[1]	pa[2] = pa[1]+p[2]	pa[3]=pa[2]+p[3]	pa[4]=pa[3]+p[4]	pa[5]=pa[4]+p[5]	pa[6]=pa[5]+p[6]
0,00000	0,10000	0,50000	0,90000	1,00000	1,00000

Considerando o vetor pa, a escolha do próximo centroide utiliza a seleção proporcional, um sorteio entre [0,1] é realizado, suponha que a o valor sorteado seja 0,15, esse valor é comparado com cada elemento do vetor de probabilidade acumulada até encontrar qual a primeira instância que tem a maior valor de probabilidade associado, na terceira comparação, x_3 é a instancia associada ao vetor com maior

probabilidade acumulada, portando, x_3 é escolhida como o terceiro centroide. O conjunto X, com a escolha do terceiro centroide, está mostrado na Figura 4.3 e o vetor de centroides na sequência.

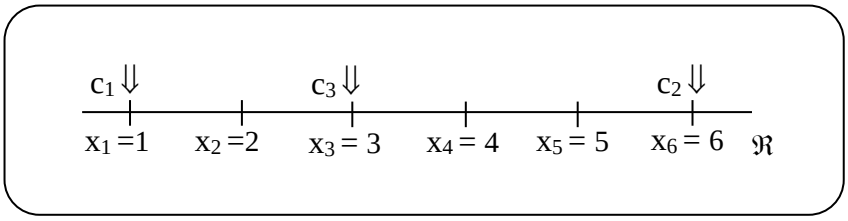


Figura 4.3 Conjunto X de instâncias de dados, após a inicialização dos três centroides (k=3).

C[1]	C[2]	C[3]
1 (x_1)	6 (x_6)	3 (x_3)

Ao usar os valores das distâncias ao quadrado, instâncias que têm uma distância pequena ao centroide que lhes é mais próximo têm baixa probabilidade de serem selecionadas, enquanto que aquelas que têm uma distância grande ao centroide que lhes é mais próximo, têm alta probabilidade de serem selecionadas.

4.1.2 Segundo Exemplo do Processo de Inicialização do k-Means++ (Espaço Bidimensional)

Considere o conjunto de nove instâncias bidimensionais, X = {x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9}, mostrado na Figura 4.4 e suponha que a tarefa de agrupamento seja a de agrupar as nove instâncias em k=3 grupos.

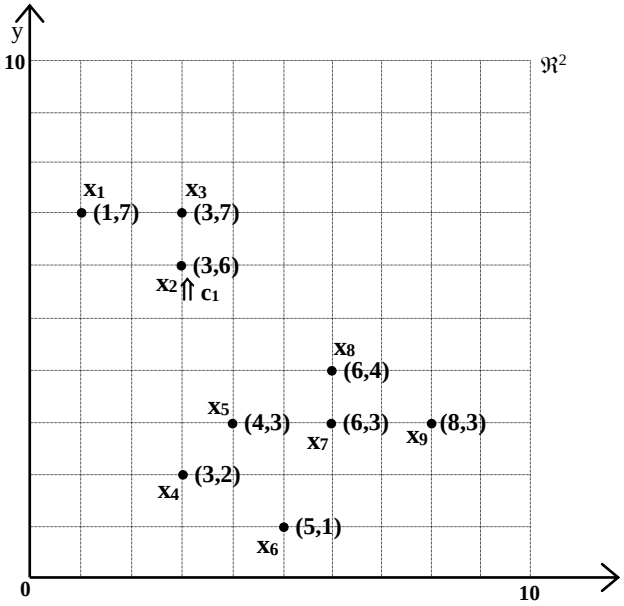


Figura 4.4 Foco na inicialização do k-Means++. Conjunto X de instâncias de dados, para serem agrupadas em k=3 grupos. A instância marcada com flecha corresponde ao primeiro centroide (c_1), escolhido randomicamente a partir do conjunto de instâncias.

As instâncias têm as seguintes coordenadas: x_1=(1,7), x_2=(3,6), x_3=(3,7), x_4=(3,2), x_5=(4,3), x_6=(5,1), x_7=(6,3), x_8=(6,4), x_9=(8,3). Neste exemplo, devido ao fato que a vírgula será usada como separador entre as duas coordenadas que definem uma instância, quando da representação de números reais com parte fracionária, o ponto será usado como separador entre a parte inteira e a parte fracionária do número.

Considere que a chamada ao procedimento *random_choice*(X), no passo 1 do Algoritmo 4.3 tenha como resultado a instância x_2. Portanto, o vetor de centroides terá a configuração:

C[1]	C[2]	C[3]
x_2 = (3,6)	–	–

Note que, até o momento, apenas um centroide foi escolhido. Considerando agora o passo do Algoritmo 4.3,

8. d[i] ← *distance_to_nearer_centroid*(x_i,C) % distância ao quadrado

o procedimento *distance_to_nearer_centroid* vai calcular, para cada uma das 9 instâncias em X, sua distância ao quadrado ao centroide que lhe seja mais próximo, construindo assim um vetor de distâncias ao quadrado, de dimensão igual a N = 9 (quantidade de elementos de X). Como existe até o momento apenas 1 centroide, c_1, serão calculadas as distâncias ao quadrado de cada uma das instâncias ao c_1 = C[1] = (3,6), resultando no vetor de distâncias ao quadrado (d), mostrado a seguir.

$d[1] = d^2(x_1,c_1)$	$d[2] = d^2(x_2,c_1)$	$d[3] = d^2(x_3,c_1)$	$d[4] = d^2(x_4,c_1)$	$d[5] = d^2(x_5,c_1)$
$2^2 + 1 = 5.00$	0.00	1.00	$4^2 = 16.00$	$3^2 + 1 = 10.00$
$d[6] = d^2(x_6,c_1)$	$d[7] = d^2(x_7,c_1)$	$d[8] = d^2(x_8,c_1)$	$d[9] = d^2(x_9,c_1)$	
$5^2 + 2^2 = 29.00$	$3^2 + 3^2 = 18.00$	$3^2 + 2^2 = 13.00$	$5^2 + 3^2 = 34.00$	

Somando todos os N (= 9) elementos do vetor d resulta em soma = 5.00 + 0.00 + 1.00 + 16.00 + 10.00 + 29.00 + 18.00 + 13.00 + 34.00 = 126.00 e, portanto, após o passo 11 do Algoritmo 4.3, dado por:

11. **for** i ← 1 to N **do** p[i] ← d[i]/soma

tem-se que o vetor de probabilidades, p, associado ao vetor de distâncias, é:

p[1] = d[1]/126.00	p[2] = d[2]/126.00	p[3] = d[3]/126.00	p[4] = d[4]/126.00	p[5] = d[5]/126.00
5.00/126.00 = 0.0396	0.00/126.00 = 0.0000	1.00/126.00 = 0.0079	16.00/126.00 = 0.1269	10.00/126.00 = 0.0793
p[6] = d[6]/126.00	p[7] = d[7]/126.00	p[8] = d[8]/126.00	p[9] = d[9]/126.00	
29.00/126.00 = 0.2301	18.00/126.00 = 0.1428	13.00/126.00= 0.1031	34.00/126.00= 0.2698	

O chamado *vetor de probabilidade acumulada*, pa, associado ao vetor p é dado por:

pa[1] = p[1]	pa[2] = pa[1] + p[2]	pa[3] = pa[2] + p[3]	pa[4] = pa[3] + p[4]	pa[5] = pa[4] + p[5]
0.0396	0.0396	0.0475	0.1744	0.2537
pa[6] = pa[5] + p[6]	pa[7] = pa[6] + p[7]	pa[8] = pa[7] + p[8]	p[9] = d[9]/126.00	
0.4838	0.6256	0.7287	$0.9985 \cong 1.0$	

Considerando o vetor pa, a escolha do próximo centroide utiliza a seleção proporcional, um sorteio entre [0,1] é realizado, suponha que o valor sorteado seja 0.73, esse valor é comparado com cada elemento do vetor de probabilidade acumulada até encontrar qual a primeira instância tem maior valor de probabilidade associado, na nona comparação, x_9 é a instancia associada ao vetor com maior probabilidade acumulada, portando, x_9 é escolhida como o segundo centroide. O conjunto X, com a escolha do segundo centroide, está mostrado na Figura 4.5 e o vetor de centroides na sequência.

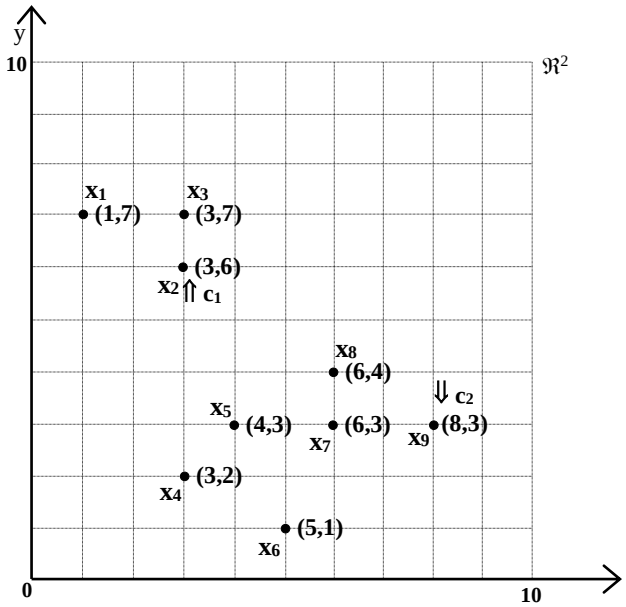


Figura 4.5 Foco na inicialização do k-Means++, após a escolha do segundo centroide, $c_2 = x_9 = (8,3)$.

C[1]	C[2]	C[3]
$x_2 = (3,6)$	$x_9 = (8,3)$	–

Até o momento dois centroides, $c_1 = (3,6)$ e $c_2 = (8,3)$ já foram escolhidos, faltando apenas a escolha de mais um para terminar o processo de inicialização, uma vez que k=3. Considerando novamente o passo 8 do Algoritmo 4.3,

8. d[i] ← *distance_to_nearer_centroid*(x_i,C) % distância ao quadrado

o procedimento *distance_to_nearer_centroid* vai calcular, para cada uma das 9 instâncias em X, sua distância ao quadrado ao centroide que lhe seja mais próximo, construindo assim um vetor de distâncias ao quadrado, de dimensão igual a N = 9 (número de elementos de X).

Como até o momento dois centroides, c_1 e c_2, foram criados, serão calculadas as distâncias ao quadrado de cada uma das nove instâncias àquele centroide que lhe for mais próximo, dentre os dois, resultando no vetor de distâncias ao quadrado (d), mostrado a seguir.

$d[1] = d^2(x_1,c_1)$	$d[2] = d^2(x_2,c_1)$	$d[3] = d^2(x_3,c_1)$	$d[4] = d^2(x_4,c_1)$	$d[5] = d^2(x_5,c_1)$
$2^2 + 1 = 5.00$	0.00	1.00	$4^2 = 16.00$	$3^2 + 1^2 = 10.00$
$d[6] = d^2(x_6,c_2)$	$d[7] = d^2(x_7,c_1)$	$d[8] = d^2(x_8,c_1)$	$d[9] = d^2(x_9,c_1)$	
$3^2 + 2^2 = 13.00$	$2^2 = 4.00$	$2^2 + 1^2 = 5.00$	0.00	

Somando todos os N (= 9) elementos do vetor d resulta em soma = 5.00 + 0.00 + 1.00 + 16.00 + 10.00 + 13.00 + 4.00 + 5.00 + 0.00 = 54.00 e, portanto, após o passo 11 do Algoritmo 4.3, dado por:

11. **for** i ← 1 to N **do** p[i] ← d[i]/soma

tem-se que o vetor de probabilidades, p, associado ao vetor de distâncias, é:

p[1] = d[1]/54.00	p[2] = d[2]/54.00	p[3] = d[3]/54.00	p[4] = d[4]/54.00	p[5] = d[5]/54.00
5.00/54.00 = 0.0925	0.00/54.00 = 0.0000.	1.00/54.00 = 0.0185	16.00/54.00 = 0.2962	10.00/54.00 = 0.1851
p[6] = d[6]/54.00	p[7] = d[7]/54.00	p[8] = d[8]/54.00	p[9] = d[9]/54.00	
13.00/54.00 = 0.2407	4.00/54.00 = 0.0740	5.00/54.00= 0.0925	0.00/54.00= 0.0000	

O chamado *vetor de probabilidade acumulada*, pa, associado ao vetor p é dado por:

pa[1] = p[1]	pa[2] = pa[1] + p[2]	pa[3] = pa[2] + p[3]	pa[4] = pa[3] + p[4]	pa[5] = pa[4] + p[5]
0.0925	0.0925	0.1110	0.4072	0.5923
pa[6] = pa[5] + p[6]	pa[7] = pa[6] + p[7]	pa[8] = pa[7] + p[8]	p[9] = d[9]/126.00	
0,833	0.907	$0.9995 \cong 1.0$	$0.9995 \cong 1.0$	

Considerando o vetor pa, a escolha do próximo centroide utiliza a seleção proporcional, um sorteio entre [0,1] é realizado, suponha que o valor sorteado seja 0.17, esse valor é comparado com cada elemento do vetor de probabilidade acumulada até encontrar qual a primeira instância tem maior valor de probabilidade associado, na quarta comparação, x_4 é a instância associada ao vetor com maior probabilidade acumulada, portando, x_4 é escolhida como o terceiro centroide. O conjunto X, com a escolha do terceiro centroide, está mostrado na Figura 4.6 e o vetor de centroides na sequência.

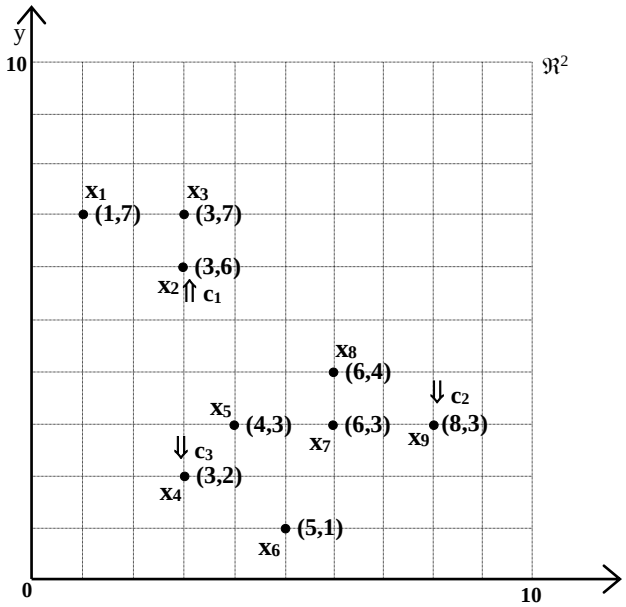


Figura 4.6 Foco na inicialização do k-Means++, após a escolha do terceiro centroide, $c_3 = x_4 = (3,2)$.

C[1]	C[2]	C[3]
$x_2 = (3,6)$	$x_9 = (8,3)$	$X_4 = (3,2)$

Como k=3 centroides foram escolhidos, o processo de inicialização termina.

4.2 O Algoritmo SPSS (*Single Pass Seed Selection*)

No artigo [Pavan *et al.* 2010] os autores criticam o algoritmo k-Means++ [Arthur & Vassilvitskii 2007], pelo fato do processo de inicialização dos centroides escolher o primeiro centroide randomicamente o que, via de regra, altera o número de iterações a cada execução e dá origem a resultados diferentes, para um mesmo conjunto de instâncias a serem agrupadas.

Os autores em [Pavan *et al.* 2011] afirmam que, "no k-Means++ a instância será escolhida com probabilidade proporcional à sua distância mínima aos centroides já escolhidos." Acrescentam, também, "que devido à seleção randômica do primeiro centroide e da seleção probabilística dos centroides restantes, diferentes execuções devem ser realizadas para obter um bom resultado. " Como pode ser evidenciado na própria definição do critério de seleção, o k-Means++ realmente determina, associada à cada instância *p*, a menor distância de *p* aos centroides já selecionados, com vistas a definir, então, sua probabilidade associada de seleção, inversamente proporcional à tal distância.

No mesmo artigo Pavan e co-autores propõem um algoritmo, que chamam de *Single Pass Seed Selection Algorithm for k-Means* (SPSS), em que o processo de inicialização do k-Means++ é ligeiramente modificado, por meio:

(1) da escolha do primeiro centroide ser feita de maneira determinística e não sujeita à uma escolha randômica, como no k-Means++ e
(2) da escolha da distância máxima que separa os centroides ser baseada na instância que esteja mais próxima de outras instâncias, no conjunto de dados.

De acordo com os autores a proposta do SPSS teve foco em selecionar o conjunto inicial de centroides de maneira a afetar positivamente:

(1) a qualidade dos grupos,
(2) o número de iterações e
(3) o número de cálculos de distância requeridos para a solução final.

Algoritmo 4.4 apresenta o pseudocódigo do processo de inicialização do k-Means++ (Algoritmo 4.3), como proposto e descrito em [Pavan *et al.* 2010] e, por essa razão, nomeado aqui de *initializaton_k-Means++_Pavan*. O pseudocódigo em Algoritmo 5 reflete exatamente aquele publicado na referência [Pavan *et al.* 2010].

procedure *initialization_k-Means++_Pavan(X,k,C)*
input: $X = \{x_1, x_2, ..., x_N\}$ % conjunto de N instâncias.
k % número de grupos.
output: $C = \{C_1, C_2, ..., C_k\}$ % conjunto de k instâncias de X escolhidos como centroides iniciais.

begin
1. Escolha uniformemente e de maneira randômica uma instância de $X = \{x_1, x_2, ..., x_N\}$ e adicione-a a C
2. Para cada instância x_i, estabeleça $D(x_i)$ como a distância entre x_i e o centroide seu mais próximo em C.
3. Escolha randomicamente um número real y de maneira uniforme entre 0 e $D(x_1)^2 + D(x_2)^2 + ... + D(x_N)^2$
4. Encontre o único número inteiro i tal que:
5. $D(x_1)^2 + D(x_2)^2 + ... + D(x_{i-1})^2 < y \leq D(x_1)^2 + D(x_2)^2 + ... + D(x_i)^2$
6. Adicione x_i a C
7. Repita os passos 2–5 até que tenham sido escolhidos k centroides.

end
return $C = \{C_1, C_2, ..., C_k\}$
end procedure

Algoritmo 4.4 Reescrita do algoritmo de inicialização de centroides do k-Means++, como apresentado em [Pavan *et al.* 2010], nomeado aqui como *initialization_k-Means++_Pavan.*

Considere novamente o Primeiro Exemplo (Seção 4.2.1), em que o conjunto de instâncias é $X = \{x_1, x_2, x_3, x_4, x_5, x_6\}$. Naquela seção foi feito um *trace* alto nível do processo de inicialização do conjunto inicial de centroides, a ser usado pelo algoritmo k-Means, para induzir um agrupamento com k=3 grupos.

No que segue será também feito um *trace* do processo de inicialização relativo ao mesmo exemplo, seguindo a notação adotada em [Pavan *et al.* 2010]. A ideia ao fazer essa repetição é para promover o entendimento, quando da apresentação do algoritmo SPSS, uma vez que as descrições dos procedimentos são diferenciadas entre si. Suponha que o passo 1 do Algoritmo 4.4 tenha também escolhido a instância $x_1 = 1$ como o primeiro centroide, resultando no vetor de centroides:

C[1]	C[2]	C[3]
$x_1 = 1$	–	–

O passo 2 do Algoritmo 4.4 produz o vetor das distâncias ao quadrado, de todas as instâncias de X ao centroide x_1 que, na notação utilizada no Algoritmo 4.4 é dado por:

$D[x_1]^2 = D[1]^2$	$D[x_2]^2 = D[2]^2$	$D[x_3]^2 = D[3]^2$	$D[x_4]^2 = D[4]^2$	$D[x_5]^2 = D[5]^2$	$D[x_6]^2 = D[6]^2$
0	1	4	9	16	25

A Figura 4.7 mostra um diagrama evidenciado as distâncias acumuladas das instâncias ao centroide mais próximo (no caso, apenas o c_1), para promover o

entendimento do passo 5 do Algoritmo 4.4. O passo 3 do Algoritmo 4.4 escolhe randomicamente um número real entre 0 e 55. Suponha que o número escolhido tenha sido o 34 que, na Figura 4.7 está apontado pela flecha. Os passos seguintes do algoritmo são:

4. Encontre o único número inteiro i tal que:
5. $D(x_1)^2 + D(x_2)^2 + ... + D(x_{i-1})^2 < y \leq D(x_1)^2 + D(x_2)^2 + ... + D(x_i)^2$

Observe na Figura 4.7 que $D(x_1)^2 + D(x_2)^2 + D(x_3)^2 + D(x_4)^2 + D(x_5)^2 < y \leq D(x_1)^2 + D(x_2)^2 + D(x_3)^2 + D(x_4)^2 + D(x_5)^2 + D(x_6)^2$, uma vez que a primeira soma é 30 e a segunda, 55. Ou seja, $30 < y \leq 55$. O número inteiro i buscado (passo 4 acima) é 6, pois o índice do x_6 e, portanto, no passo 6, o segundo centroide passa a ser x_6.

Note que se o número y escolhido randomicamente fosse, por exemplo, 6, a desigualdade expressa no passo 5 do algoritmo seria dada por $D(x_1)^2 + D(x_2)^2 < y \leq D(x_1)^2 + D(x_2)^2 + D(x_3)^2$ e, portanto, o número inteiro i buscado seria o 4 e o segundo centroide, c_2, seria então o x_4.

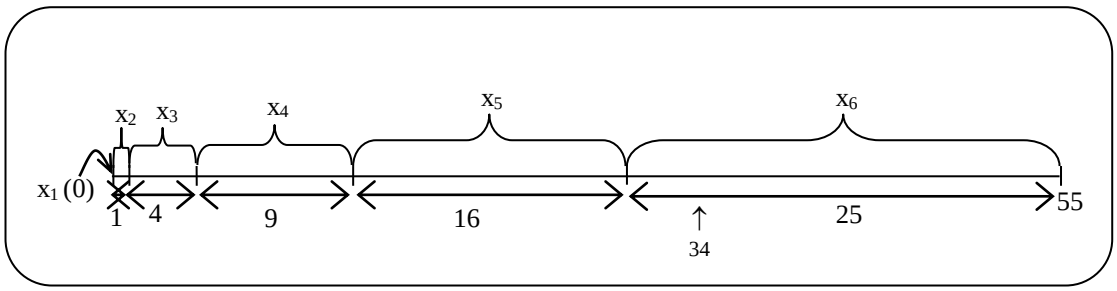


Figura 4.7 Diagrama com as distâncias ao quadrado acumuladas, considerando todas as instâncias do conjunto X. O total das distâncias ao quadrado é 55. Valor randomicamente escolhido y=34 mostrado com a flecha.

Conforme apresentado e discutido em [Pavan et *al.* 2010], dado um conjunto $X = \{x_1, x_2, x_3, ..., x_N\}$ com N instâncias, o algoritmo SPSS escolhe um conjunto $C = \{C_1, C_2, ..., C_k\}$, de k centroides iniciais, a partir de X.

No SPSS o primeiro centroide é inicializado com uma instância de X que esteja mais perto do maior número possível de instâncias em X. A seguir, assumindo que as N instâncias de X estão distribuídos uniformemente em k (número de grupos) grupos, estima-se que cada grupo contenha n/k instâncias. São então calculadas as somas das distâncias da instância selecionada (primeiro centroide) às primeiras n/k instâncias e essa soma é considerada como y.

A descrição completa do SPSS, que cria um conjunto C de k centroides, a partir do conjunto de N instâncias $X = \{x_1, x_2, x_3, ..., x_N\}$ é apresentada em pseudocódigo no Algoritmo 4.5.

A notação empregada na descrição do algoritmo segue de perto aquela apresentada na referência [Pavan *et al.* 2010]. A opção por manter a notação original nesta dissertação se deve ao fato do SPSS ter sido um algoritmo que recebeu críticas, em virtude de não ser bem compreendido na comunidade científica voltada à investigação de algoritmos de agrupamentos, principalmente devido à sua descrição ser ambígua, o que permite diferentes interpretações do processo.

procedure *Single_Pass_Seed_Selection(X,k,C)*
input: $X = \{x_1, x_2, ..., x_N\}$ % conjunto de N instâncias.
k % número de grupos.
output: $C = \{C_1, C_2, ...,C_k\}$ % conjunto de k instâncias de X escolhidas como centroides iniciais.

begin
1. Calcule a matriz de distância $Dist_{N\times N}$, de maneira que Dist[i,j] seja a distância entre as instâncias x_i e x_j.
2. Encontre Sumv, de maneira que Sumv[i] seja a soma das distâncias da instância x_i a todas as demais instâncias de X.
3. Encontre o indice h tal que valor(x_h) = min(Sumv) e faça indice $\leftarrow$ h
4. $C \leftarrow \{x_{indice}\}$
5. Para cada instância x_i, calcule $D(x_i)$ como a distância entre x_i e o centroide seu mais próximo em C.
6. Faça y a soma das distâncias das primeiras n/k instâncias mais próximas de x_{indice}
7. Encontre o único número inteiro i tal que:
8. $D(x_1)^2 + D(x_2)^2 + ... + D(x_{i-1})^2 < y \leq D(x_1)^2 + D(x_2)^2 + ... + D(x_i)^2$
9. $C \leftarrow C \cup \{x_i\}$
7. Repita os passos 5–9 até que tenham sido escolhidos k centroides.
end
return C
end procedure

Algoritmo 4.5 Algoritmo *Single Pass Seed Selection* (SPSS) para inicialização dos centroides.

Considere novamente o conjunto $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$, mostrado na Figura 4.4 e que está reapresentado na Figura 4.8, em que as instâncias têm as seguintes coordenadas: x_1=(1,7), x_2=(3,6), x_3=(3,7), x_4=(3,2), x_5=(4,3), x_6=(5,1), x_7=(6,3), x_8=(6,4), x_9=(8,3).

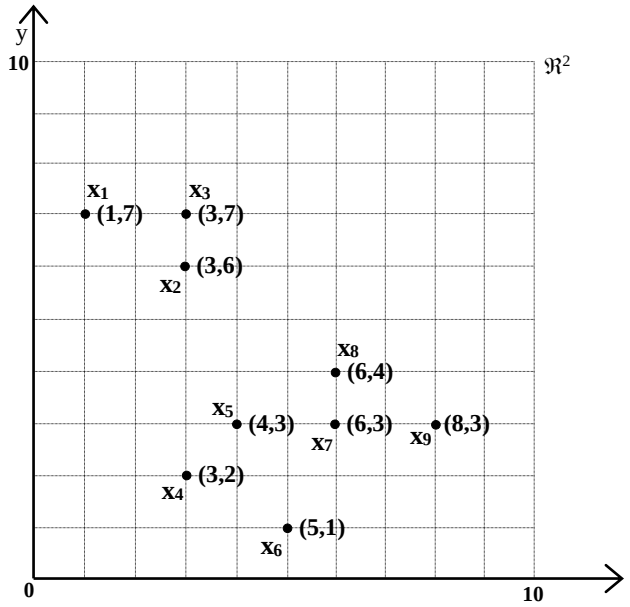


Figura 4.8 Conjunto de 9 instâncias de dados usadas como exemplos para a criação do conjunto inicial de centroides, para k=3.

No que segue será mostrado um trace do Algoritmo 4.5, para a construção do conjunto inicial de centroides, como proposto em [Pavan *et al.* 2010]. Suponha que a tarefa de agrupamento seja a de agrupar as nove instâncias em k=3 grupos.

$$\text{Dist} = \begin{pmatrix} 0 & 5 & 4 & 29 & 25 & 52 & 41 & 34 & 65 \\ 5 & 0 & 1 & 16 & 10 & 29 & 18 & 13 & 34 \\ 4 & 1 & 0 & 25 & 17 & 40 & 25 & 18 & 41 \\ 29 & 16 & 25 & 0 & 2 & 5 & 10 & 13 & 26 \\ 25 & 10 & 17 & 2 & 0 & 5 & 4 & 5 & 16 \\ 52 & 29 & 40 & 5 & 5 & 0 & 5 & 10 & 13 \\ 41 & 18 & 25 & 10 & 4 & 5 & 0 & 1 & 4 \\ 34 & 13 & 18 & 13 & 5 & 10 & 1 & 0 & 5 \\ 65 & 34 & 41 & 26 & 16 & 13 & 4 & 5 & 0 \end{pmatrix} \quad \text{Sumv} = \begin{pmatrix} 255 \\ 126 \\ 171 \\ 126 \\ 84 \\ 159 \\ 108 \\ 99 \\ 204 \end{pmatrix}$$

O passo 3 do Algoritmo 4.5 determina o valor de ***índice***:

3. Encontre o ***indice*** tal que valor(x_{indice}) = min(Sumv) e encontrando a instância de maior densidade x_{indice}.

Considerando o vetor Sumv, valor(x_5) = 84 = min(Sumv) e ***índice*** = 5 e x_5 é a instância com "maior densidade" termo empregado pelo autor [Pavan *et al.* 2010], (*i.e.*, instâncias com maior quantidades de instâncias próximas). O primeiro centroide é então escolhido como $c_1 = x_5$. A seguir o passo 5 do Algoritmo 4.5 é realizado e consiste em:

5. Para cada instância x_i, calcule d(x_i) como a distância (quadrada) entre x_i e o centroide seu mais próximo em C.

que resultará em:

$d[1] = d^2(x_1,c_1)$	$d[2] = d^2(x_2,c_1)$	$d[3] = d^2(x_3,c_1)$	$d[4] = d^2(x_4,c_1)$	$d[5] = d^2(x_5,c_1)$
$4^2 + 3^2 = 25.00$	$3^2 + 1^2 = 10.00$	$4^2 + 1^2 = 17.00$	$1^2 + 1^2 = 2.00$	0.00
$d[6] = d^2(x_6,c_1)$	$d[7] = d^2(x_7,c_1)$	$d[8] = d^2(x_8,c_1)$	$d[9] = d^2(x_9,c_1)$	
$2^2 + 1^2 = 5.00$	$2^2 = 4.00$	$2^2 + 1^2 = 5.00$	$4^2 = 16.00$	

O passo 6 instrui para:

6. Faça y a soma das distâncias das primeiras n/k (9/3 = 3) instâncias mais próximas de $x_{índice}$.

As 3 instâncias mais próximas de x_{indice} são: x_4, x_7 e x_6 e, portanto, y = 2.00 + 4.00 + 5.00 = 11.00. Nos passos 7 e 8:

7. Encontre o único número inteiro i tal que:
8. $D(x_1)^2 + D(x_2)^2 + ... + D(x_{i-1})^2 < 11 \leq D(x_1)^2 + D(x_2)^2 + ... + D(x_i)^2$

o valor de i encontrado é i=1 e, portanto, $C = \{c_1 = x_5, c_2 = x_1\}$; $h \leftarrow 1$

O passo 5 do Algoritmo 4.5 é realizado novamente, resultando em:

$d[1] = d^2(x_1,c_2)$	$d[2] = d^2(x_2,c_2)$	$d[3] = d^2(x_3,c_2)$	$d[4] = d^2(x_4,c_1)$	$d[5] = d^2(x_5,c_1)$
0.00	$2^2 + 1^2 = 5.00$	$2^2 = 4.00$	$1^2 + 1^2 = 2.00$	0.00
$d[6] = d^2(x_6,c_1)$	$d[7] = d^2(x_7,c_1)$	$d[8] = d^2(x_8,c_1)$	$d[9] = d^2(x_9,c_1)$	
$2^2 + 1^2 = 5.00$	$2^2 = 4.00$	$2^2 + 1^2 = 5.00$	$4^2 = 16.00$	

O passo 6 instrui para:

6. Faça y a soma das distâncias das primeiras n/k (9/3 = 3) instâncias mais próximas de x_h.

As 3 instâncias mais próximas de $c_1 = x_5$ são: x_4, x_7 e x_6 e, portanto, y = 2.00 + 4.00 + 5.00 = 11. Nos passos 7 e 8:

7. Encontre o único número inteiro i tal que:
8. $D(x_1)^2 + D(x_2)^2 + D(x_3)^2 < 11.00 \leq D(x_1)^2 + D(x_2)^2 + D(x_3)^2 + D(x_4)^2$

o valor encontrado de i é i=4 e, portanto, $C = \{ c_1 = x_5, c_2 = x_1, c_3 = x_4\}$.

Capítulo 5

Algoritmos de Inicialização do k-Means: *CCIA*, *Method-1* e *Maedeh-Suresh*

Este capítulo aborda três algoritmos de inicialização de centroides, todos eles com pretensões de melhorar o desempenho do k-Means, com vistas tanto a contribuir com a convergência do processo iterativo, quanto com a qualidade do agrupamento induzido pelo algoritmo k-Means, principal foco deste trabalho de pesquisa.

Na Seção 5.1 é abordado o algoritmo *Cluster Center Inicialization Algorithm* (CCIA) proposto em [Khan & Ahmad 2004]; na Seção 5.2 é abordado o algoritmo *Method-1*, proposto em [Al-Daoud & Roberts 1996] e, finalmente, na Seção 5.3 é abordado o algoritmo *Design of Efficient k-Means Clustering Algorithm with Improved Initial Centroids*, proposto em [Maedeh & Suresh 2013] que é referenciado nesta dissertação como algoritmo Maedeh-Suresh.

5.1 *Cluster Center Initialization Algorithm (CCIA)*

De acordo com Khan & Ahmad [Khan & Ahmad 2004] o algoritmo CCIA foi proposto com vistas a obter uma boa inicialização de centroides, para ser usada pelo k-Means, logo no início de seu processo de indução do agrupamento pretendido. O algoritmo CCIA foi baseado em duas observações associadas a processos de agrupamento sob o k-Means:

(1) a de que algumas instâncias são muito semelhantes entre si e, devido a isso, elas acabam fazendo parte de um mesmo grupo, independentemente da maneira como seja feita a escolha inicial dos centroides de grupos;

(2) atributos, individualmente, podem também fornecer algumas informações a respeito dos centroides iniciais de grupos. O CCIA visa

exclusivamente a inicialização dos centroides para o k-Means e se aplica apenas a dados numéricos.

Os autores justificam a falta de comparações de resultados com outros algoritmos de inicialização de centroides (que não o randômico) informando, com base em comentários encontrados em [Meila & Heckerman 1998], que não há métodos universalmente aceitos para selecionar centroides de grupos iniciais. Khan & Ahmad em [Khan & Ahmad 2004] comparam os resultados do CCIA com a proposta original do k-Means, em que centroides iniciais são escolhidos randomicamente, utilizando cinco conjuntos de instâncias de dados.

5.1.1 Pseudocódigo e Detalhes de Funcionamento do *CCIA*

O Algoritmo 5.1 exibe o pseudocódigo do CCIA, como proposto e descrito em [Khan & Ahmad 2004], procurando manter a maior fidelidade possível à descrição original. O procedimento *Merge_DBMSDC*, utilizado pelo CCIA, está apresentado em Algoritmo 5.2. As mudanças introduzidas no pseudocódigo foram apenas as relativas a nomes de variáveis e estruturas, com o objetivo de manter as convenções de notação adotadas neste trabalho. O algoritmo assume uma distribuição normal para os valores de cada um dos atributos que descrevem as instâncias de dados. O CCIA espera como entrada:

(1) o conjunto de instâncias $X = \{x_1, x_2, ..., x_N\}$, em que cada instância de dado x_i ($i = 1, ..., N$) é descrita por M valores, cada um deles associado a cada um dos M atributos, $A_1, A_2, ..., A_M$, que descrevem as instâncias; e

(2) o número de grupos, representado pela variável k.

O algoritmo produz, como saída, um conjunto de k centroides, como resultado da fase de inicialização do k-Means. O CCIA focaliza um atributo por vez, desde o primeiro até o M-ésimo.

procedure CCIA*(X,k,S)*
input: X % conjunto de N instâncias.
k % número de grupos.
output: S = {CG_1, CG_2, ..., CG_k}

begin
Para cada atributo A_j (j = 1, ..., M) considerado repita os passos 1-8
(1) calcular a média μ_j e o desvio padrão σ_j associados às instâncias descritas apenas pelo atributo A_j.
(2) calcular o percentil z_s, que corresponde à área sob a curva normal, de $-\infty$ a z_s, definida como $((2\times s) - 1)/2\times k$, para s = 1, 2, ..., k.
(3) calcular o valor do atributo correspondente a cada percentil, usando a média e o desvio padrão dos atributos como $x_s = z_s \times \sigma_{j+} \mu_j$.
(4) criar um agrupamento inicial com base na distância euclidiana entre os x_s e e as instâncias de X descritas apenas pelo atributo A_j;
(5) Executar o k-Means no conjunto X descrito apenas por A_j, tendo como agrupamento inicial o criado em (4).
(6) Alocar os rótulos de grupos obtidos em (5) a cada uma das instâncias.
(7) Executar o k-Means usando o conjunto X descrito por todos os atributos;
(8) Armazenar o rótulo do grupo ao qual cada instância pertence;
(9) Gerar a cadeia de rótulos associada a cada um dos padrões - ao final cada padrão terá M rótulos associados, cada um deles associado a um determinado atributo;
(10) Encontrar cadeias de rótulos distintas, que correspondem ao número de grupos distintos, representado por k', em que $k' \geq k$.
(11) Encontrar o centroide de cada um dos k' grupos.
(12) Se $k' > k$, usar o algoritmo *Merge-DBMSDC* (Algoritmo 5.2) aos k' grupos para obter k grupos;
(13) Unir aqueles grupos cujos centroides estão ocorrendo no mesmo conjunto. Como existem k conjuntos de padrões, serão obtidos k grupos.
(14) Encontre a média dos padrões pertencentes a cada um dos k grupos para obter os k centroides iniciais (CG_1, CG_2, ..., CG_k) que serão usados como centroides iniciais do k-Means.
end
return S = {CG_1, CG_2, ...,CG_k}
end procedure

Algoritmo 5.1 Pseudocódigo do CCIA, para identificar os centroides que serão utilizados para a fase de inicialização do k-Means.

O Algoritmo 5.2 descreve de maneira superficial e sem os detalhes necessários, o processo de junção de grupos em um agrupamento, implementado pelo algoritmo Density-Based Multi Scale Data Condensation [Mitra *et al.* 2002].

procedure *Merge-DBMSDC(k',k,S)*
input: k': k'(>k) sequência de m grupos rotulados para cada instância chamada "cadeia de instâncias".
k: número de grupos.
output: S = {CG_1, CG_2, ...,CG_k}

begin
% processo de inicialização da fusão de grupos
(1) seja k' o número de grupos gerado pelo algoritmo CCIA e k'>k
(2) calcular o centro de grupo para cada um dos k' grupos
(3) considerar B = {CG_1, CG_2, ..., $CG_{k'}$}o conjunto com k' centros de grupos
(4) escolher um inteiro positivo q=1, inicializar i ← 1 e repita os passos (5-10) até B = ∅
(5) para cada centroide $CG_i \in B$, calcular a distância ao seu q-ésimo vizinho mais próximo em B, denotada por $r_{q,CGi}$
(6) selecionar um centroide $CG_j \in B$ com o menor valor de $r_{q,CGj}$. Empates com relação ao valor mais baixo podem ser resolvidos seguindo alguma convenção, tal como escolher o menor dentre os índices dos centroides empatados, etc.
(7) criar um conjunto $S_i = \varnothing$
(8) adicionar CG_j ao conjunto S_i
(9) remover todos os centroides de B que estejam dentro de um círculo com centro em CG_j e raio $1.5 \times r_{q,CGj.}$ e os inclua em S_i. O conjunto B, composto pelos centroides restantes deve ser renomeado como B.
(10) atualizar i: i ← i + 1
% finalização do processo de fusão de grupos K'
end
return S = { CG_1, CG_2, ...,CG_k}
end procedure

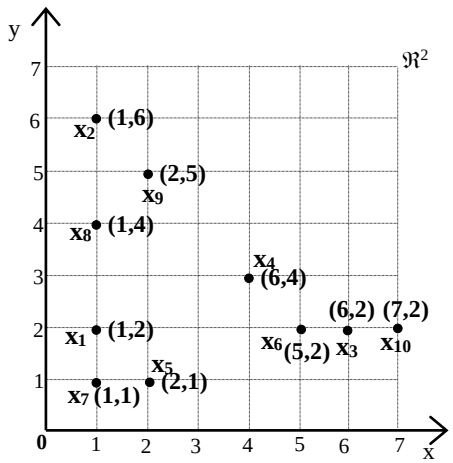
Algoritmo 5.2 Pseudocódigo do algoritmo *Merge-DBMSDC.*

5.1.2 Um Diagrama Geral do Funcionamento do *CCIA*

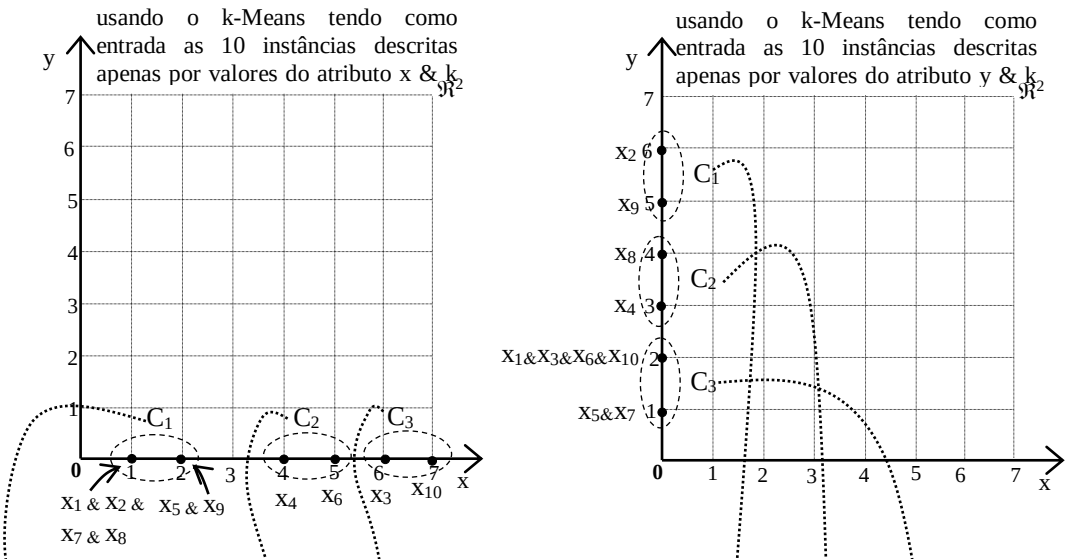
Com o objetivo de promover o entendimento do pseudocódigo do CCIA, esta seção exibe uma situação hipotética em que o conjunto de instâncias de dados, para ser agrupado em k=3 grupos, é constituído por N=10 instâncias de dados, exibidas na Tabela 5.1 e mostradas nos gráficos das próximas figuras desta seção.

Tabela 5.1 Conjunto de instâncias bidimensionais, descritas pelos atributos x e y, a serem agrupadas.

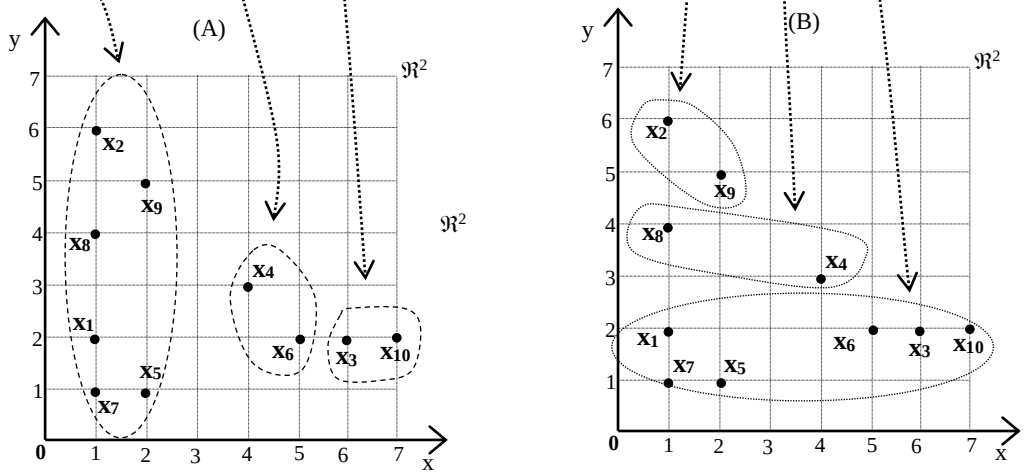
Instância	Atributo x	Atributo y
x_1	1	2
x_2	1	6
x_3	6	2
x_4	4	3
x_5	2	2
x_6	5	2
x_7	1	1
x_8	1	4
x_9	2	5
x_{10}	7	2



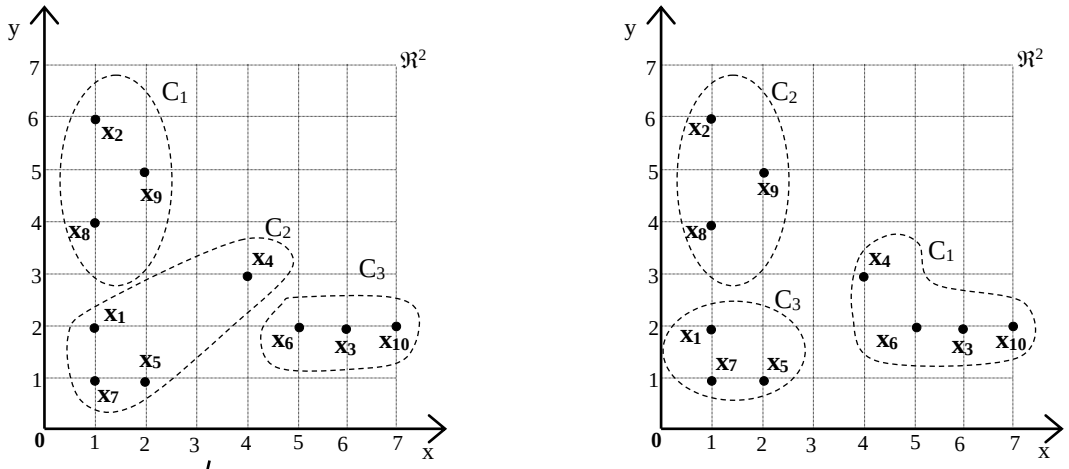
(I) AGRUPAMENTO POR ATRIBUTO



(II) USANDO OS GRUPOS DEFINIDOS EM (I) COMO INICIALIZAÇÃO AO K_MEANS PARA AGRUPAR AS INSTÂNCIAS DESCRITAS USANDO TODOS OS ATRIBUTOS



(III) RESULTADO DO PROCESSO DE AGRUPAMENTO CONDUZIDO PELO K-MEANS, A PARTIR DAS INICIALIZAÇÕES EM (II), (A) E (B)



Instância	Grupo (foco atributo x)	Grupo (foco atributo y)	Notação da instância
x_1	2	3	x_1–23
x_2	1	2	x_2 –12
x_3	3	1	x_3–31
x_4	2	1	x_4–21
x_5	2	3	x_5–23
x_6	3	1	x_6–31
x_7	2	3	x_7–23
x_8	1	2	x_8–12
x_9	1	2	x_9–12
x_{10}	3	3	x_{10}–31

Grupos formados com foco em x e com foco em y (última coluna da tabela anterior)	Instâncias que participam do grupo/(nro. delas)
23	x_1, x_5, x_7/(3)
12	x_2, x_8, x_9/(3)
31	x_3, x_6, x_{10}/(3)
21	x_4/(1)

Uma vez obtido o agrupamento via o CCIA, como tal agrupamento tem 4 grupos, a condição do Algoritmo 5.1 no passo (12) é satisfeita, tal agrupamento é passado ao procedimento de junção de grupos, que implementa o algoritmo Density-Based Multi Scale Data Condensation (DBMSDC), Algoritmo 5.2, para a junção de dois grupos em um, a redução do número de grupos de 4 para 3. A instância X_4 é mesclada ao terceiro grupo. A média do agrupamento encontrado S = {{ x_1, x_5, x_7},{ x_2, x_8, x_9}, {x_3, x_6, x_{10},

x_4}} é utilizado como centroides iniciais para k-Means. A junção entre o grupo 3 e 4 é assumida visto que o centroide da instância x_4, está dentro do raio estabelecido pelo algoritmo DBMSDC.

5.2 O Algoritmo *Method-1*

Essa seção apresenta e discute o primeiro de dois algoritmos de inicialização para o k-Means, propostos em [Al-Daoud & Roberts 1996]. O algoritmo, identificado por *Method-1* na referência anterior, pode ser caracterizado como baseado em *grid*, uma vez que assume que o espaço definido pelos dados esteja dividido em um determinado número de células, todas elas com a mesma dimensão. O processo de inicialização dos centroides é feito de uma vez só e, de acordo com os autores, não há a necessidade de outras tentativas para a definição dos centroides iniciais. O número de células sugerido pelos autores do algoritmo é $\frac{k}{2\sqrt{k}}$, onde k é o número de grupos. O pseudocódigo alto nível do algoritmo Method-1 é exibido em Algoritmo 5.3.

Algoritmo Method-1

Assuma que o espaço seja dividido em células de mesmo tamanho
Seja:
N: o número de instâncias de dados
k: número de grupos desejados
NC_i: número de instâncias de dados na célula i
M: número de células

1. Para cada célula calcule a parte inteira de $KC_i = NC_i \times k/N$;
2. Selecione randomicamente KC_i instâncias na célula i como centroides iniciais das instâncias de dados na célula i;
3. Se $\Sigma_{i=1,\ldots M} KC_i = k$, PARE
4. Se $\Sigma_{i=1,\ldots M} KC_i < k$ seja $DEF = k - \Sigma_{i=1,\ldots M} KC_i$.
   Se DEF for um número alto, digamos, > 10% de k, faça $M = M - (sqrt(M) - 1)^2$ e volte ao passo 1.
   Caso contrário, selecione randomicamente DEF instâncias de dados. PARE.

Algoritmo 5.3 Pseudocódigo do algoritmo Method-1 [Al-Daoud & Roberts 1996].

Também, como comentam os autores, o algoritmo não requer um limite inferior em relação ao número de instâncias de dados. O Method-1 está fundamentado na ideia de inicializar os centros de grupos de acordo com a distribuição das instâncias de dados, em nível macro, e deixar a tarefa de agrupamento propriamente dita, a cargo do algoritmo de agrupamento que, espera-se, melhore e refine a solução inicial fornecida pelo Method-1. O algoritmo distribui os centroides de maneira direta, direcionado pela densidade das instâncias de dados. Como o algoritmo tem problemas e sua descrição está incompleta, esse documento optou por incluir seu pseudocódigo próximo de como publicado na referência [Al-Daoud & Roberts 1996], com exceção de alguns nomes de variáveis, que se mantiveram iguais aos adotados nesse trabalho (*e.g.*, k para indicar o

número de grupos) e dos termos em língua inglesa, que foram traduzidos para o português.

Na notação das coordenadas de cada instância, o ponto é usado como o separador entre as partes inteira e fracionária; a vírgula é o separador entre as duas coordenadas.

5.2.1 O Algoritmo *Method-1* – Primeiro Exemplo

O conjunto de instâncias de dados X utilizado no exemplo está mostrado na Figura 5.1. O espaço bidimensional, definido pelas nove instâncias do conjunto X, particionado em 25 células que compõem o grid, e são , está mostrado na Figura 5.2.

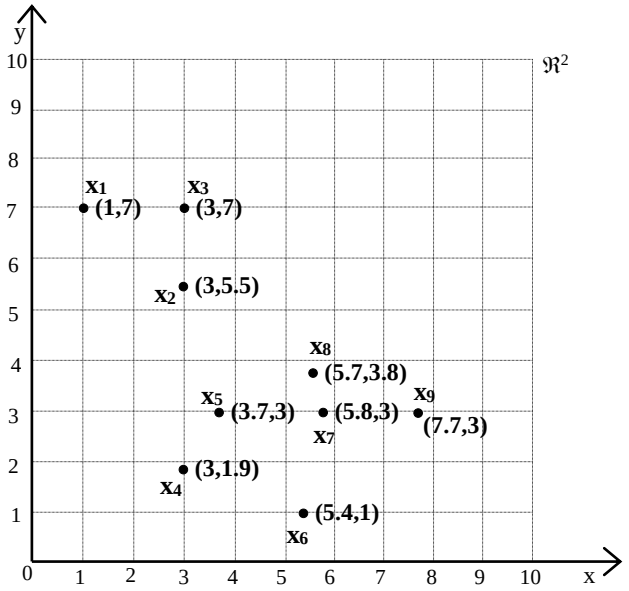


Figura 5.1 Conjunto de instâncias iniciais X.

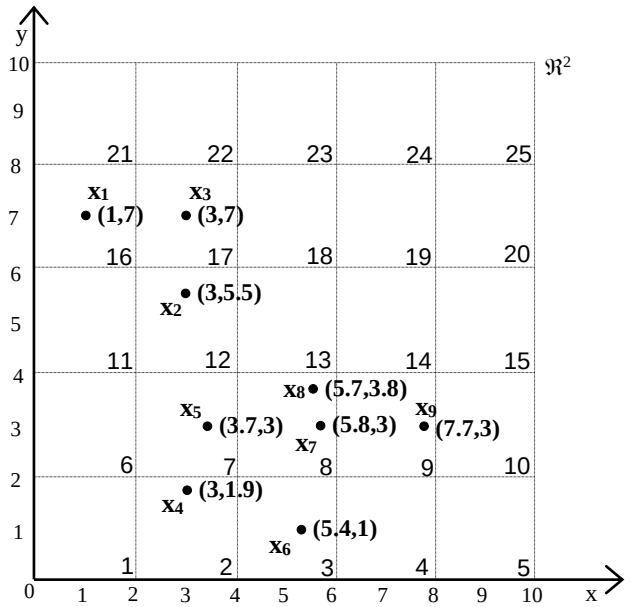


Figura 5.2 Espaço das 9 instâncias de dados, dividido igualmente em M=25 células de mesmo tamanho.

No que segue, a Tabela 5.2 mostra o número de instâncias em cada uma das 25 células do grid. A Tabela 5.3 mostra o cálculo de KC_i (i=1,...,25), que considera o número de instâncias de dados total, o número de instâncias de dados por célula e o número de grupos (k).

Tabela 5.2 Número de instâncias em cada uma das M=25 células da Figura 5.3.

Número de instâncias de dados por célula				
$NC_1 = 0$	$NC_6 = 0$	$NC_{11} = 0$	$NC_{16} = 1$	$NC_{21} = 0$
$NC_2 = 1$	$NC_7 = 1$	$NC_{12} = 1$	$NC_{17} = 1$	$NC_{22} = 0$
$NC_3 = 1$	$NC_8 = 2$	$NC_{13} = 0$	$NC_{18} = 0$	$NC_{23} = 0$
$NC_4 = 0$	$NC_9 = 1$	$NC_{14} = 0$	$NC_{19} = 0$	$NC_{24} = 0$
$NC_5 = 0$	$NC_{10} = 0$	$NC_{15} = 0$	$NC_{20} = 0$	$NC_{25} = 0$

Tabela 5.3 Resultado do procedimento *parte_inteira*, considerando os valores de NC_i (i=1, ..., M=25) (Tabela 5.2), k = 3 (número de grupos) e N=9, número de instâncias de dados. Tal procedimento calcula o número de centroides/célula a ser considerado.

$KC_i \leftarrow$ *parte_inteira*($NC_i \times k/N$)	
$KC_1 = NC_1 \times k/N = 0 \times 3/9 = 0$	$KC_{14} = 0$
$KC_2 = NC_2 \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0	$KC_{15} = 0$
$KC_3 = NC_3 \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0	$KC_{16} = NC_{16} \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0
$KC_4 = 0$	$KC_{17} = NC_{17} \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0
$KC_5 = 0$	$KC_{18} = 0$
$KC_6 = 0$	$KC_{19} = 0$
$KC_7 = NC_7 \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0	$KC_{20} = 0$
$KC_8 = NC_8 \times k/N = 2 \times 3/9 = 0.666 =$ *parte_inteira*(0.666) = 0	$KC_{21} = 0$
$KC_9 = NC_9 \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0	$KC_{22} = 0$
$KC_{10} = 0$	$KC_{23} = 0$
$KC_{11} = 0$	$KC_{24} = 0$
$KC_{12} = NC_{12} \times k/N = 1 \times 3/9 = 0.333 =$ *parte_inteira*(0.333) = 0	$KC_{25} = 0$
$KC_{13} = 0$	

Note que quando da execução do comando da linha 3 do Algoritmo 5.3, considerando os valores exibidos nas tabelas 5.2 e 5.3, o valor $\Sigma_{i=1, \ldots M} KC_i = 0 < k$, o que faz com que a variável DEF seja calculada como k – 0 = k, implicando, então, uma escolha randômica dos k centroides iniciais, tornando o uso do algoritmo inócuo. Essa situação evidencia um problema com a proposta do Method-1, que é o de não levar em conta o número de células a ser considerado.

A existência desse problema foi corroborada por autores da publicação [He *et al.* 2004]. O texto que segue neste parágrafo é uma paráfrase do que eles dizem. Os centroides iniciais são inicializados localmente, em cada sub-espaço correspondente (*i.e.*, o da célula), de maneira randômica. Apesar desse método ter funcionado bem com um valor relativamente alto para k em um estudo anterior [Al-Daoud & Roberts 1994], ele é altamente dependente de uma escolha bastante cuidadosa do valor de M. Quando o valor de k é pequeno, M deve ser pequeno o suficiente, de maneira que alguns subespaços possam ter algum centroide inicial associado.

O exemplo mostrado nessa seção evidenciou o problema apontado na literatura. O próximo exemplo usa um número menor de células, considerando que o valor de k é pequeno *i.e.*, 3.

5.2.2 O Algoritmo *Method-1* – Segundo Exemplo

A Figura 5.3 apresenta o mesmo conjunto de N=9 instâncias de dados anterior, com uma diferente escolha de M, M=4.

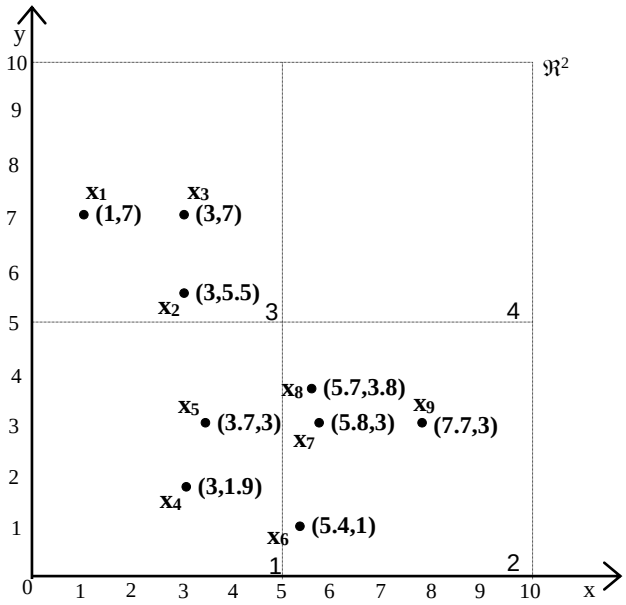


Figura 5.3 Espaço das instâncias dividido igualmente em M=4 células de mesmo tamanho.

Tabela 5.4 Número de instâncias em cada uma das M=4 células da Figura 5.3.

Número de instâncias de dados por célula
$NC_1 = 2$
$NC_2 = 4$
$NC_3 = 3$
$NC_4 = 0$

Tabela 5.5 Resultado do procedimento *parte_inteira*, considerando os valores de NC_i (i=1, ..., M=4) (Tabela 5.4), k = 3 (número de grupos) e N=9, número de instâncias de dados. Tal procedimento calcula o número de centroides/célula a ser considerado.

$K_{C_i} \leftarrow$ *parte_inteira*($NC_i \times k/N$)
$KC_1 = NC_1 \times k/N = 2 \times 3/9 = 0.666 =$ *parte_inteira*(0.666) = 0
$KC_2 = NC_2 \times k/N = 4 \times 3/9 = 1.333 =$ *parte_inteira*(1.333) = 1
$KC_3 = NC_3 \times k/N = 3 \times 3/9 = 1.000 =$ *parte_inteira*(1.000) = 1
$KC_4 = NC_4 \times k/N = 0 \times 3/9 = 0$

Considerando os resultados mostrados na Tabela 5.5, dois centroides iniciais são randomicamente escolhidos: 1 na célula C_2 e outro na C_3; por exemplo, $CG_1 = x_7$ e $CG_2 = x_3$. Como $\Sigma_{i=1, ...4} KC_i < 4$ ($\Sigma_{i=1, ...4} KC_i = 2$), seja DEF = k – $\Sigma_{i=1, ...M} KC_i$ (DEF = 3 – 2 = 1). No caso, o valor de DEF sendo 1 (que não é um número alto mas satisfaz o critério

de número alto sugerido no algoritmo *i.e.*, é maior que 0,3), o algoritmo prossegue alterando o valor do número de células para M = 4 – (sqrt(4) – 1)^2 = 3 e o algoritmo recomeça novamente. Caso o valor 1 não seja considerado um número alto, apesar de satisfazer o critério dos autores do que seja um número alto, o algoritmo simplesmente seleciona randomicamente DEF (=1) instância de dados para ser o terceiro centroide, o que intuitivamente parece ser o procedimento mais óbvio a ser feito.

5.2.3 O Algoritmo *Method-1* – Terceiro Exemplo

A Figura 5.4 apresenta o um novo conjunto de N=14 instâncias de dados criado pelo autor dessa dissertação, escolha de M, M=4.

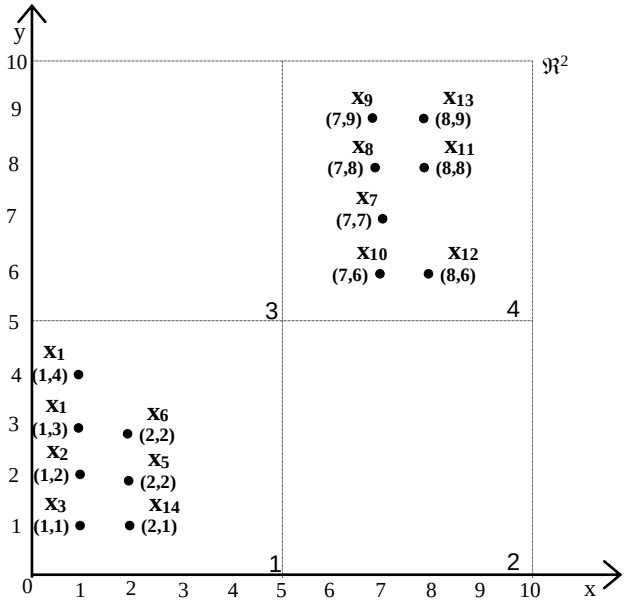


Figura 5.4 Espaço das instâncias dividido igualmente em M=4 células de mesmo tamanho.

Tabela 5.6 Número de instâncias em cada uma das M=4 células da Figura 5.4.

Número de instâncias de dados por célula
$NC_1 = 7$
$NC_2 = 0$
$NC_3 = 0$
$NC_4 = 7$

Tabela 5.7 Resultado do procedimento *parte_inteira*, considerando os valores de NC_i (i=1, ..., M=4) (Tabela 5.6), k = 2 (número de grupos) e N=14, número de instâncias de dados. Tal procedimento calcula o número de centroides/célula a ser considerado.

$KC_i \leftarrow$ *parte_inteira*($NC_i \times k/N$)
$KC_1 = NC_1 \times k/N = 7 \times 2/14 = 1 =$ *parte_inteira*(1.000) = 1
$KC_2 = NC_2 \times k/N = 0 \times 2/14 = 0$
$KC_3 = NC_3 \times k/N = 0 \times 2/14 = 0$
$KC_4 = NC_4 \times k/N = 7 \times 2/14 = 1 =$ *parte_inteira*(1.000) = 1

Considerando os resultados mostrados na Tabela 5.7, dois centroides iniciais são randomicamente escolhidos: 1 na célula C_1 e outro na C_4; por exemplo, $CG_1 = x_5$ e $CG_2 = x_{11}$. Como $\Sigma_{i=1,\ldots 4} KC_i = 2$, seja $KC_i = k$, **PARE**. No caso, o valor de KC_i é igual a K o critério de parada do Algoritmo 5.3 no passo (3) é satisfeito, então, as instâncias { x_5, x_{11}} são escolhidas como centroides iniciais mostrados na Tabela 5.8.

Tabela 5.8 Padrões do conjunto original que foram escolhidos como os 2 centroides iniciais (k=2).

C[1]	C[2]
X_5=2,2	X_{11}=8,8

Para teste do algoritmo Method-1 foi necessário criar o conjunto de dados mostrado na Figura 5.4. Infelizmente os autores do algoritmo, em sua publicação não descrevem um conjunto de instâncias para teste, também não citaram a fonte do conjuntos de dados por eles utilizado, o que não permite comparações empíricas de resultados com a implementação realizada.

5.3 O Algoritmo Maedeh-Suresh

O algoritmo k-Means em sua proposta original [MacQueen 1967], escolhe seus centroides de forma randômica; esta escolha randômica, via de regra, evita que o algoritmo induza o mesmo agrupamento, quando executado novamente, com o mesmo conjunto de instâncias.

No artigo [Maedeh & Suresh 2013] os autores propõem uma mudança no algoritmo original k-Means, por meio da alteração da escolha dos centroides iniciais. Afirmam que o algoritmo proposto por eles tem maior acurácia e, quando do uso do mesmo conjunto de instâncias, tem a vantagem de produzir sempre o mesmo agrupamento. O algoritmo proposto pelos autores é dividido em três procedimentos, descritos por Algoritmo 5.4, Algoritmo 5.5 e Algoritmo 5.6.

```
procedure Maedeh_Suresh (X,k,AG)
input: X  % conjunto de X instâncias.
       k  % número desejado de grupos.
output: AG = {G1, G2, ..., Gk} % conjunto de k grupos.

begin
% processo que encontra os centroides iniciais (C) usando o Algoritmo 5.5.
call  Maedeh_Suresh_step1(X,k,C)
% processo que associa cada padrão de X ao grupo adequado, usando o Algoritmo 5.6.
call  Maedeh_Suresh_step2(X,C,AG)
% final do algoritmo.
end
return AG = {G1, G2, ..., Gk}
end procedure
```

Algoritmo 5.4 Pseudocódigo do algoritmo Maedeh_Suresh [Maedeh & Suresh 2013], com chamadas aos algoritmos (1) *Maedeh_Suresh_step1*(*X,k,C*), para escolha dos centroides, e (2) *Maedeh_Suresh_step2*(*X,C,AG*), para atribuição de instâncias aos respectivos grupos.

.

O processo de inicialização proposto pelos autores é baseado na escolha de bons centroides. Inicialmente o algoritmo proposto calcula a distância de todas as instâncias de dados do conjunto, à origem do plano cartesiano e, então, escolhe como primeiro centroide a instância que tem a menor distância à origem do plano cartesiano e, como segundo centroide, aquela instância que tem a maior distância à origem do plano cartesiano. Os próximos centroides são escolhidos calculando a maior distância entre as distâncias mais próximas, entre as instâncias e centroides já escolhidos, até que o número de centroides escolhidos seja igual a k. O algoritmo proposto é dividido em três procedimentos, Algoritmo 5.4, Algoritmo 5.5 e Algoritmo 5.6.

O pseudocódigo mostrado no Algoritmo 5.4 inicia com uma chamada ao procedimento *Maedeh_Suresh_step1* (Algoritmo 5.5), que calcula os centroides iniciais e, na sequência, faz uma chamada ao procedimento *Maedeh_Suresh_step2* (Algoritmo 5.6), utilizando como parâmetros de entrada os centroides encontrados no passo anterior, que associa cada instância ao grupo apropriado.

```
procedure  Maedeh_Suresh_step1(X,k,C)

input: X  % conjunto de N instâncias.
       k  % número de grupos.
output: C = {c1, c2, ..., ck} % conjunto de centroides encontrados.

  begin
    (1) para cada instância de X  calcular a sua distância da origem do plano cartesiano
    (2) ordenar as distâncias obtidas no passo (1) e ordenar as instâncias de acordo com suas distâncias
    (3) escolher a primeira e última instâncias, como o primeiro c1 e segundo c2 centroides iniciais.
    (4) para cada instância xi de X calcular sua distância aos centroides já escolhidos;
    (5) associar cada instância xi ao grupo cujo centroide lhe seja mais próximo.
    (6) para cada instância xi
      (6.1) atribuir GrupoId[i] = j
      (6.2) atribuir DistanciaProxima[i] = d(xi,cj)
    (7) escolher a instância com máxima distância no vetor DistanciaProxima, como o próximo
        centroide inicial.
    (8)  repetir os passos (4),(5),(6) e (7) até cj = k
  end
return C = {c1, c2, ...,ck}
end procedure
```

Algoritmo 5.5 Pseudocódigo do procedimento *Maedeh_Suresh_step1* [Maedeh & Suresh 2013], que encontra os centroides iniciais.

```
procedure  Maedeh_Suresh_step2(X,C,AG)

input:   X                          % conjunto de N instâncias.
         C= {c1, c2, c3...cn}       % k centroides iniciais encontrado no Algoritmo 5.5.
output: AG = {G1, G2, ..., Gk}   % conjunto de agrupamentos.

  begin
   (1) calcular a distância entre cada instância xi ao centroide cj
   (2) para cada instância xi encontrar o centroide cj mais próximo dela e associa-lo ao grupo j
   (3) atribuir GrupoId[i] = j
   (4) atribuir DistanciaProxima[i] = d(xi,cj)
   (5) para cada grupo j recalcular os centroides.
   (6) para cada instância xi:
        (6.1) calcular as distancias do centroide para o presente grupo mais próximo.
        (6.2) se esta distância é menor ou igual a distância atual, o padrão permanece no mesmo grupo;
              senão
                  (6.2.1) para cada centroide cj calcular a distância d(di,cj)
                  (6.2.2) associar a instância xi com o grupo mais próximo do centroide cj
                  (6.2.3) atribuir GrupoId [i] = j
                  (6.2.4) atribuir DistanciaProxima[i] = d(xi,cj)
   (7) para cada grupo j (1<=j<=k), realcular os centroides.
   (8) repetir os passos (6 e 7) até que o critério de convergência seja satisfeito
  end
return AG = {G1, G2, ..., Gk}
end procedure
```

Algoritmo 5.6 Pseudocódigo do procedimento *Maedeh_Suresh_step2* [Maedeh & Suresh 2013] para a atribuição adequada de cada instância ao seu grupo.

Na notação das coordenadas de cada instância, o ponto é usado como o separador entre as partes inteira e fracionária; a vírgula é o separador entre as duas coordenadas.

5.3.1 Exemplo de Uso do Algoritmo *Maedeh-Suresh*

No exemplo apresentado nesta seção foi utilizado o mesmo conjunto de 30 instâncias de dados (nomeado X), utilizado no experimento descrito em [Maedeh & Suresh 2013]. O uso do mesmo conjunto foi intencional, com o objetivo de comparar resultados apresentados na referência em que o algoritmo foi proposto, e aqueles obtidos pela implementação do CCIA, parte do sistema computacional desenvolvido durante a realização do trabalho de pesquisa descrito nesta dissertação.

A Tabela 5.9 lista as coordenadas das 30 instâncias do conjunto X = {x_1, x_2, x_3, ..., x_{30}}, e a Figura 5.5 mostra as 30 instâncias no plano cartesiano.

Tabela 5.9 Conjunto de instâncias bidimensionais utilizadas com o algoritmo Maedeh-Suresh.

x_1=(2,1.7)	x_6=(29.6,28)	x_{11}=(66,65)	x_{16}=(62,62.63)	x_{21}=(11,19)	x_{26}=(49,50)
x_2=(14,15.8)	x_7=(23,22.6)	x_{12}=(26,25)	x_{17}=(68,66.3)	x_{22}=(16,16)	x_{27}=(48,46.67)
x_3=(1,1)	x_8=(3.2,2.5)	x_{13}=(0.3,0.6)	x_{18}=(81,80)	x_{23}=(84,84)	x_{28}=(19,20)
x_4=(15,15)	x_9=(5,4)	x_{14}=(1.5,1)	x_{19}=(81.4,74)	x_{24}=(88,88)	x_{29}=(55.7,43)
x_5=(30,31)	x_{10}=(65.8,57)	x_{15}=(1.67,1.5)	x_{20}=(83.6,85)	x_{25}=(50,55)	x_{30}=(17,18.23)

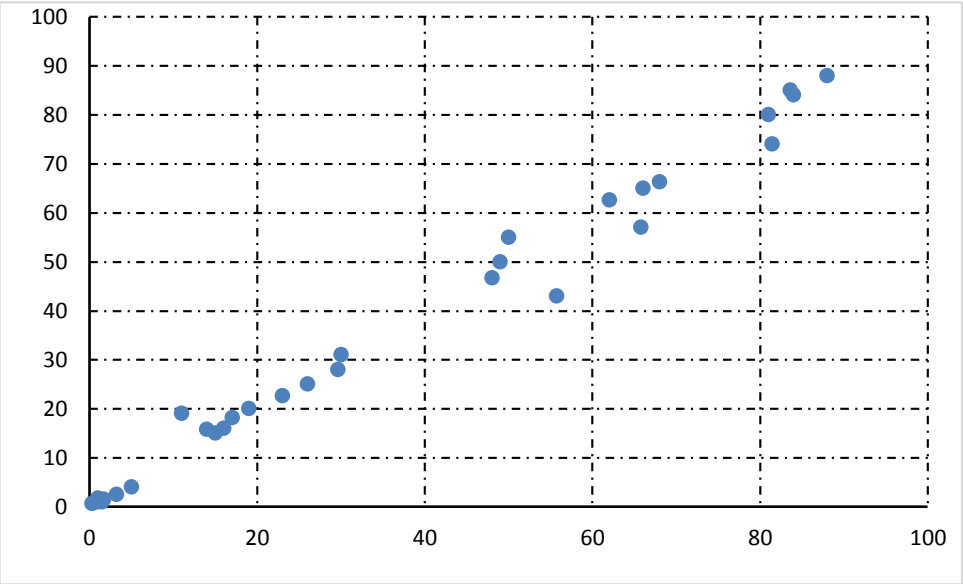


Figura 5.5 Conjunto com 30 instâncias de dados utilizadas como entrada para o algoritmo Maedeh-Suresh.

Como o valor de k utilizado no experimento descrito em [Maedeh & Suresh 2013] não foi informado no texto do artigo, o valor de k = 4 foi inferido a partir da figura (incorreta) publicada no artigo e mostrada na Figura 5.6. O fato dos experimentos realizados durante este trabalho não confirmarem os resultados publicados, levou a um questionamento da suposição feita *i.e.*, a de k=4. O primeiro autor do trabalho publicado foi contatado via e-mail e confirmou o erro existente na figura reproduzida na Figura 5.6, e gentilmente forneceu a figura correta, mostrada na Figura 5.7.

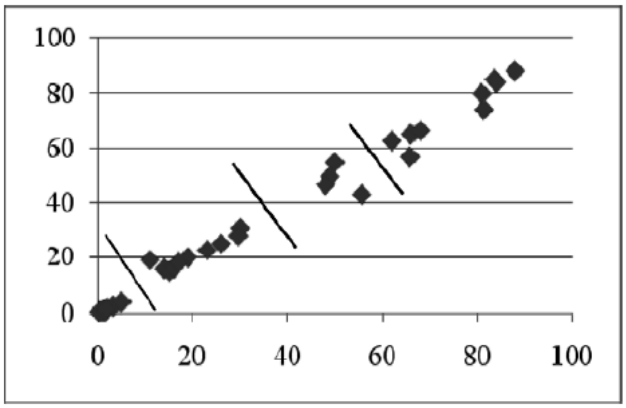


Figura 5.6 Figura (incorreta) publicada em [Maedeh & Suresh 2013], com o resultado do agrupamento induzido pelo algoritmo Maedh-Suresh.

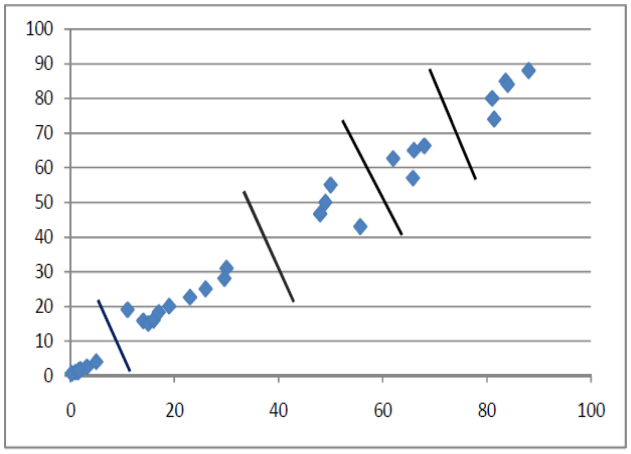


Figure 5.7: Final Five Clusters According to Proposed Algorithm

Figura 5.7 Figura correta, encaminhada pelo primeiro autor do artigo descrito em [Maedeh & Suresh 2013], com o resultado do agrupamento, com 5 grupos, induzido pelo algoritmo Maedh-Suresh.

O procedimento *Maedeh_Suresh_step1* no passo (1) calcula as distâncias de todas as instâncias à origem do plano cartesiano e, no passo (2), ordena as instâncias em ordem crescente de suas respectivas distâncias à origem. A Tabela 5.10 mostra o resultado obtido.

Tabela 5.10 Ordenação das 30 instâncias de dados em ordem crescente de suas respectivas distâncias à origem do plano cartesiano.

Ordem	Padrão	Distância	Ordem	Padrão	Distância	Ordem	Padrão	Distância
1	x_{13}	0.670	11	x_{22}	22.627	21	x_{25}	74.330
2	x_3	1.414	12	x_{30}	24.926	22	x_{10}	87.055
3	x_{14}	1.802	13	x_{28}	27.586	23	x_{16}	88.127
4	x_{15}	2.244	14	x_7	32.245	24	x_{11}	92.633
5	x_1	2.624	15	x_{12}	36.069	25	$x1_7$	94.972
6	x_8	4.060	16	x_6	40.745	26	x_{19}	110.008
7	x_9	6.403	17	x_5	43.139	27	x_{18}	113.846
8	x_2	21.110	18	x_{27}	66.948	28	x_{23}	118.793
9	x_4	21.213	19	x_{26}	70.007	29	x_{20}	119.222
10	x_{21}	21.954	20	x_{29}	70.366	30	x_{24}	124.450

No passo (3), utilizando a ordenação obtida, o algoritmo escolhe a primeira instância (*i.e.*, aquela com a menor distância à origem do plano cartesiano) como primeiro centroide (c_1) e a instância com a maior distância à origem do plano cartesiano,

como segundo centroide (c_2). No exemplo, considerando os valores apresentados na Tabela 5.10, $c_1 = x_{13}$ e $c_2 = x_{24}$.

No passo (4) o algoritmo calcula as distâncias de cada instância x_i a cada centroide já criado *i.e.*, c_1 e c_2. Os cálculos estão mostrados na Tabela 5.11. No passo 5, tais valores induzem o agrupamento com os grupos $c_1 = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{12}, x_{13}, x_{14}, x_{15}, x_{21}, x_{22}, x_{28}, x_{30}\}$ e $c_2 = \{x_{10}, x_{11}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{23}, x_{24}, x_{25}, x_{26}, x_{27}, x_{29}\}$.

Tabela 5.11 Valores das distâncias entre cada instância x_i e os centroides c_1 e c_2.

Instâncias	Vetor de Distâncias
$d(x_{1\text{-}10}, c_1)$	[2.024, 20.462, 0.806, 20.577, 42.5, 40.115, 31.611, 3.466, 5.8, 86.436]
$d(x_{11\text{-}20}, c_1)$	[91.999, 35.437, 0, 1.264, 1.639, 87.49, 94.338, 113.211, 109.383, 118.584]
$d(x_{21\text{-}30}, c_1)$	[21.284, 21.992, 118.157, 123.814, 73.684, 69.368, 66.315, 26.945, 69.763, 24.283]
Instâncias	**Vetor de Distâncias**
$d(x_{1\text{-}10}, c_2)$	[121.834, 103.386, 123.036, 103.237, 81.32, 83.729, 92.207, 120.421, 118.088, 38.129]
$d(x_{11\text{-}20}, c_2)$	[31.827, 88.391, 123.814, 122.683, 122.209, 36.326, 29.51, 10.63, 15.477, 5.325]
$d(x_{21\text{-}30}, c_2)$	[103.392, 101.823, 5.656, 0, 50.328, 54.451, 57.516, 96.876, 55.392, 99.543]

Os próximos passos de *Maedeh_Suresh_step1* atribuem à cada posição i do vetor *GrupoId*, o grupo (dentre os dois grupos até então criados) ao qual a instância x_i ($1 \leq i \leq 30$) pertence (Tabela 5.12) e atribui a cada posição do vetor *DistanciaProxima* a correspondente distância de x_i ao centroide do grupo ao qual pertence (Tabela 5.13).

Tabela 5.12 Associação de padrões a centroides mais próximos e criação do vetor *GrupoId*, com dimensão N=30.

Centroide	Padrões
c_1	$[x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{12}, x_{13}, x_{14}, x_{15}, x_{21}, x_{22}, x_{28}, x_{30}]$
c_2	$[x_{10}, x_{11}, x_{16}, x_{17}, x_{18}, x_{19}, x_{20}, x_{23}, x_{24}, x_{25}, x_{26}, x_{27}, x_{29}]$
GrupoId	[1 1 1 1 1 1 1 1 1 2 2 1 1 1 1 2 2 2 2 2 1 1 2 2 2 2 2 1 2 1]

Tabela 5.13 Vetor *DistanciaProxima*, de dimensão N=30. Cada posição i contém o valor da distância do padrão x_i ao centroide do grupo ao qual pertence (ver vetor *GrupoId* – Tabela 5.9). Em negrito estão as distâncias (zero) os padrões que representam os dois centroides criados até o momento.

i	*DistânciaProxima*[i]	i	*DistânciaProxima*[i]	i	*DistânciaProxima*[i]
1	2.624	11	31.827	21	21.284
2	20.462	12	35.437	22	21.992
3	0.806	**13**	**0.0**	23	5.656
4	20.577	14	1.264	**24**	**0.0**
5	42.5	15	1.639	25	50.328
6	40.115	16	36.326	26	54.451
7	31.611	17	29.51	27	57.516
8	3.466	18	10.63	28	26.945
9	5.8	19	15.477	28	55.392
10	38.129	20	5.325	30	24.283

No passo (7) o procedimento *Maedeh_Suresh_step1* identifica a maior distância armazenada no vetor *DistanciaProxima*. A maior distância identificada, 57.516, é aquela entre o padrão x_{27} e o centroide c_2, do grupo ao qual x_{27} pertence. Portanto, o padrão x_{27} é eleito o próximo centroide, c_3. Como o critério de parada não foi satisfeito ainda (*i.e.*, o número de centroides escolhidos até então não é 4), o algoritmo repete os passos 4 – 7 em busca do último centroide, c_4. Novamente são calculadas as distâncias entre cada padrão e os 3 centroides já escolhidos; a atribuição dos padrões aos grupos representados pelos 3 centroides está mostrada na Tabela 5.14. Os vetores *GrupoId* e *DistanciaProxima* são atualizados como na Tabela 5.14 e Tabela 5.15. A Tabela 5.16 exibe a distância de cada padrão ao centroide c_3.

Tabela 5.14 Associação de padrões a centroides mais próximos e atualização do vetor *GrupoId*, com dimensão N=30.

Centroide	Padrões
c_1	[x_1, x_2, x_3, x_4, x_7, x_8, x_9, x_{13}, x_{14}, x_{15}, x_{21}, x_{22}, x_{28}, x_{30}]
c_2	[x_{18}, x_{19}, x_{20}, x_{23}, x_{24}]
c_3	[x_5, x_6, x_{10}, x_{11}, x_{12}, x_{16}, x_{17}, x_{25}, x_{26}, x_{27}, x_{29}]
GrupoId	[1,1,1,1,3,3,1,1,1,3,3,3,1,1,1,3,3,2,2,2,1,1,2,2,3,3,3,1,3,1]

Tabela 5.15 Vetor *DistanciaProxima*, de dimensão N=30. Cada posição i contém o valor da distância do padrão x_i ao centroide do grupo ao qual pertence (ver vetor *GrupoId* – Tabela 5.11). Em negrito estão as distâncias (zero) os padrões que representam os três centroides criados até o momento.

i	DistânciaProxima[i]	i	DistânciaProxima[i]	i	DistânciaProxima[i]
1	2.624	11	25.69	21	21.284
2	20.462	12	30.88	22	21.992
3	0.806	**13**	**0.0**	23	5.656
4	20.577	14	1.264	**24**	**0.0**
5	23.865	15	1.639	25	8.566
6	26.213	16	21.23	26	3.476
7	31.611	17	28.023	**27**	**0.0**
8	3.466	18	10.63	28	26.945
9	5.8	19	15.477	29	8.529
10	20.58	20	5.325	30	24.283

Tabela 5.16 Distâncias entre cada padrão x_i ($1 \leq i \leq 30$) e o centroide c_3 (note que as distâncias entre cada padrão e os centroides c_1 e c_2 estão na Tabela 5.8).

Padrões	Distâncias
$d(x_{1\text{-}10}, c_3)$	[64.329, 45.923, 65.534, 45.738, 23.865, 26.213, 34.703, 62.912, 60.578, 20.58]
$d(x_{11\text{-}20}, c_3)$	[25.69, 30.88, 66.315, 65.176, 64.705, 21.23, 28.023, 46.902, 43.156, 52.312]
$d(x_{21\text{-}30}, c_3)$	[46.202, 44.324, 51.86, 57.516, 8.566, 3.476, 0, 39.399, 8.529, 42.069]

O procedimento *Maedeh_Suresh_step1* no passo (7) escolhe a maior distância armazenada no vetor *DistanciaProxima*, tal distância é 31.611 e está associada à distância do padrão x_7 ao centroide c_1. O padrão x_7 é então escolhido como o centroide c_4. Como o número de centroides escolhidos é igual a k (inicialmente estabelecido como 4), o Algoritmo 5.4 finaliza retornando o conjunto de centros C{c_1=x_{13}, c_2=x_{24}, c_3=x_{27}, c_4=x_7 }. Na Tabela 5.17 estão mostrados os valores associados a cada um dos quatro centroides e, na Figura 5.8, estão mostrados os 30 padrões do conjunto original e assinalados cada um dos quatro centroides escolhidos pelo procedimento *Maedeh_Suresh_step1*.

Tabela 5.17 Padrões do conjunto original que foram escolhidos como os 4 centroides iniciais (k=4).

C[1]	C[2]	C[3]	C[4]
x_{13}=0.3,0.6	x_{24}=88,88	x_{27}=48,46.67	x_7=23,22.6

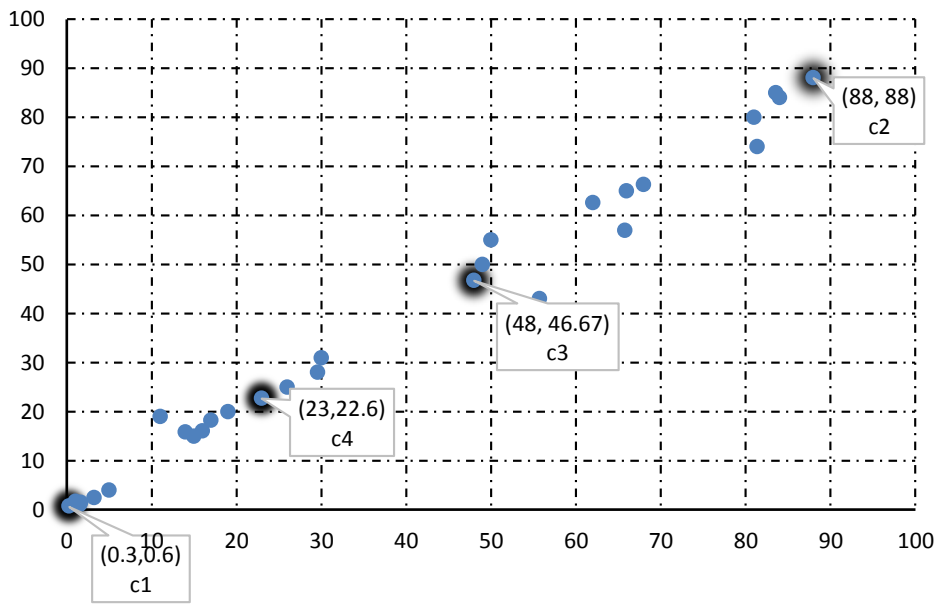


Figura 5.8 Conjunto X de 30 padrões de dados. Os padrões sombreados correspondem aos k = 4 centroides {c_1, c_2, c_3, c_4} escolhidos pelo Algoritmo 5.4 (*Maedeh_Suresh_step1).*

Capítulo 6

Descrição dos Experimentos, Avaliação dos Resultados, Conclusões do Trabalho e Trabalhos Futuros

Durante o trabalho de pesquisa descrito nesta dissertação foi projetado e desenvolvido um sistema computacional, chamado I-kMeans, com o objetivo de avaliar experimentalmente as contribuições dos algoritmos que foram objeto de pesquisa deste trabalho, ou seja, o k-Means++ (Seção 4.1), o SPSS (Seção 4.2), o CCIA (Seção 5.2), o Method-1 (Seção 5.3) e o Maedeh-Suresh (Seção 5.4).

Este capítulo está organizado como segue. A Seção 6.1 apresenta uma descrição das principais funcionalidades disponibilizadas pelo sistema I-kMeans. Na sequência, na Seção 6.2 é descrito um conjunto de experimentos realizados com os cinco algoritmos de inicialização de centroides, para uso acoplado ao k-Means, que foram apresentados e discutidos nos capítulos 4 e 5. A seção aborda os conjuntos de dados utilizados nos experimentos, a metodologia usada para a realização dos experimentos e coleta dos resultados, bem como promove uma discussão comparativa entre os algoritmos de inicialização, com base nos resultados obtidos. A Seção 6.3 finaliza o capítulo, apresentando um texto que condensa todo o trabalho de pesquisa realizado e as principais conclusões da pesquisa realizada, principalmente aquelas subsidiadas pelos resultados obtidos nos experimentos. Na Seção 6.4 finaliza o capítulo com sugestões para continuidade do trabalho.

Como comentado anteriormente, indefinições relacionadas à descrição do algoritmo SSPS (Seção 4.3) dificultaram sua implementação, exigindo um conjunto de testes para confirmar partes do pseudocódigo desse algoritmo, que estavam ambíguas compreendidas durante a implementação do algoritmo.

Problemas em relação a alguns aspectos do CCIA foram contornados por meio de explicações, dadas pelo autor do algoritmo, que foi contatado via e-mail. Já o problema apontado na Seção 5.3.1, relativo ao algoritmo Method-1, foi contornado por meio da implementação realizada permitindo o uso de conjunto de instâncias n-dimensionais, parte do sistema computacional disponibilizado, que inseriu uma pequena alteração do algoritmo publicado. No que segue serão utilizadas as seguintes abreviações: k-Means++ como ++, o SPSS como SPSS, o CCIA como CCIA, o Method-1 como M1 e o Maedeh-Suresh como MS.

6.1 Detalhes do Sistema Computacional Implementado

O sistema I-kMeans (*Initializing k-Means*) foi desenvolvido usando a linguagem de programação JAVA versão 1.7, na plataforma de desenvolvimento Eclipse versão Luna, sob o sistema operacional Linux Ubuntu versão 17.04.

O I-kMeans pode ser considerado multiplataforma ou seja, o seu desenvolvimento em Java permite que o sistema seja executado em qualquer plataforma (sistema operacional) sem necessidade de alterações no código fonte. Para a execução do sistema é necessário que a máquina virtual Java JRE (*Java Runtime Environment*) tenha sido instalada. A arquitetura do sistema I-kMeans é composta por três módulos que são descritos, na sequência, nas três próximas subseções, 6.1.1, 6.1.2 e 6.1.3.

6.1.1 Módulo de Importação e Pré-processamento de Arquivos (painel Import & Preprocess)

O sistema I-kMeans é iniciado no painel “Import & Preprocess” que, entre suas funcionalidades, permite carregar um arquivo contendo um conjunto de instâncias de dados, normalizar atributos e remover atributos com valores duplicados. A Figura 6.1 mostra o painel inicial “Import & Preprocess” do sistema I-kMeans.

O I-kMeans permite carregar um conjunto de instâncias por meio da leitura de arquivos texto que estejam em formato ARFF (*Attribute-Relation File Format*). A leitura preenche a tabela com as instâncias e seus respectivos valores de atributos. Na Figura 6.2 são mostradas as várias funcionalidades por meio de marcação e enumeração de cada item do painel “Import & Preprocess” do sistema I-kMeans.

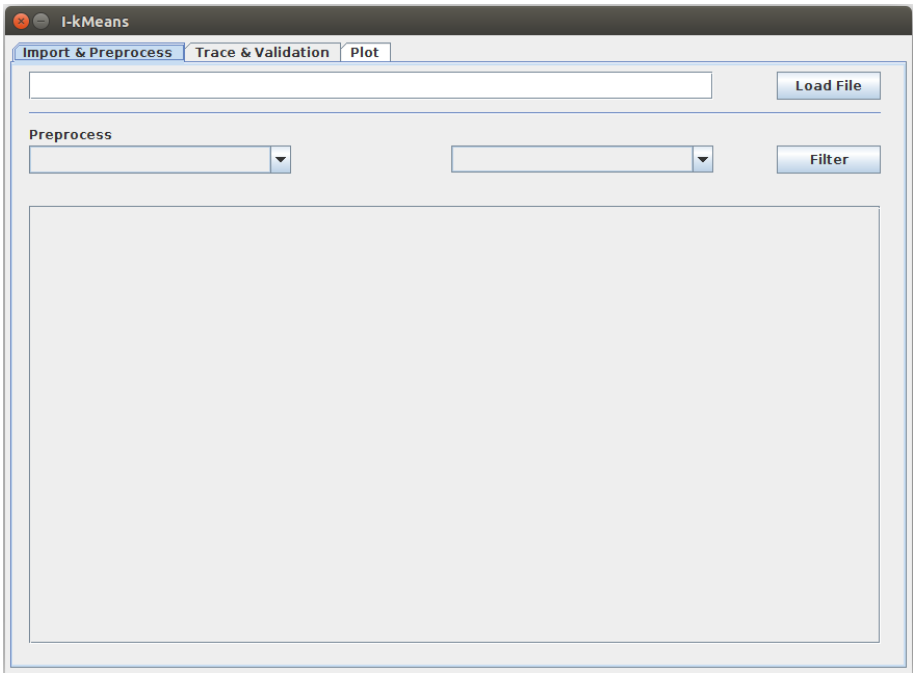


Figura 6.1 Painel “Import & Preprocess” do sistema I-kMeans.

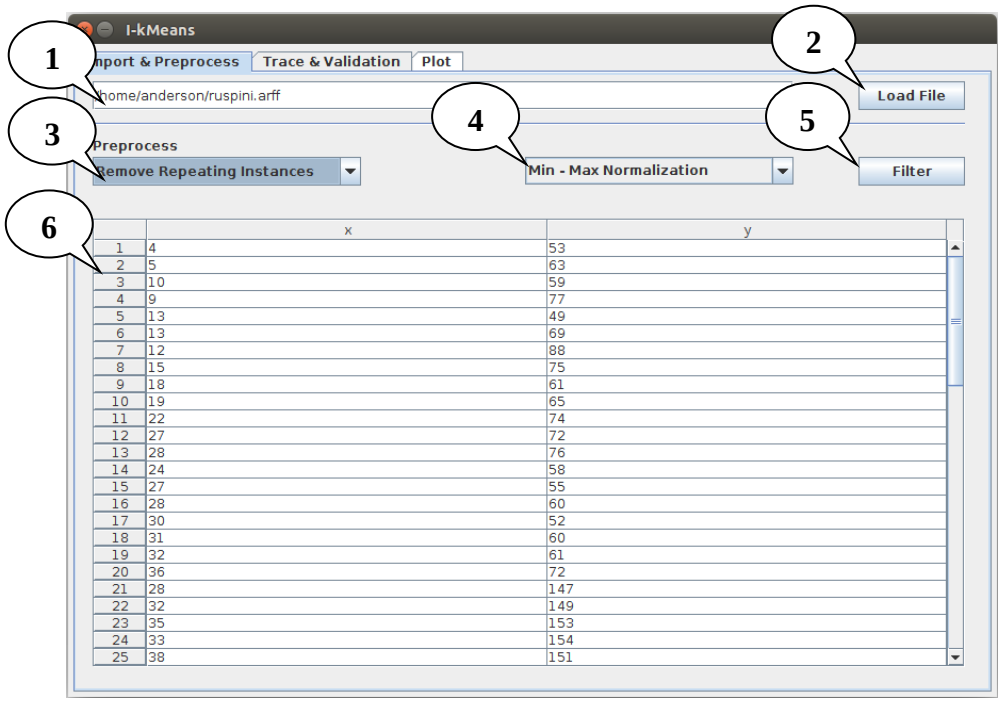


Figura 6.2 Painel “Import & Preprocess” com marcação e enumeração, para as explicações que seguem.

(1) Caixa de texto com o nome e caminho completo do arquivo a ser carregado.

(2) O botão “Load File” permite que o arquivo possa ser carregado automaticamente em memória; o carregamento do arquivo é evidenciado por meio do preenchimento da tela (ver item (6)).

(3) O *combo box* “Remove Repeating Instances” pode ser escolhido para solicitar ao sistema que remova instâncias repetidas do conjunto de instâncias carregado; se deixado em branco o conjunto de instâncias original não é alterado.

(4) O *combo box* “Min-Max Normalization” pode ser escolhido para que a normalização Min-Max seja aplicada ao conjunto de instâncias, o que permite que nenhum atributo domine o outro; se deixado em branco o conjunto de instâncias original não é alterado.

(5) O botão “Filter” permite a execução de pré-processamento dos dados, por meio das opções: “Remove Repeating Instances” e “Min-Max Normalization”. Ao acionar o botão “Filter” as instâncias são pré-processadas e a área para a exibição do conjunto de instâncias é atualizada.

(6) Área para a exibição do conjunto de instâncias. Quando o arquivo carregado possui os rótulos dos atributos, as colunas são preenchidas com rótulos informados no arquivo. Na ausência dos rótulos, eles são automaticamente criados com os nomes *default* de Attr1, Attr2, ..., AttrM, em que M é o total de atributos do conjunto de instâncias. As linhas são enumeradas iniciando em 1 até o número total de instâncias e cada linha representa uma instância.

6.1.2 Módulo de Validação e Execução do Algoritmo (painel Trace & Validation)

Este módulo permite a execução dos algoritmos implementados, identificados no início deste capítulo, bem como a visualização tanto dos resultados obtidos quanto das validações realizadas. Na Figura 6.3 as várias funcionalidades são mostradas por meio de marcação e enumeração de cada item do painel “Trace & Validation” do sistema I-kMeans.

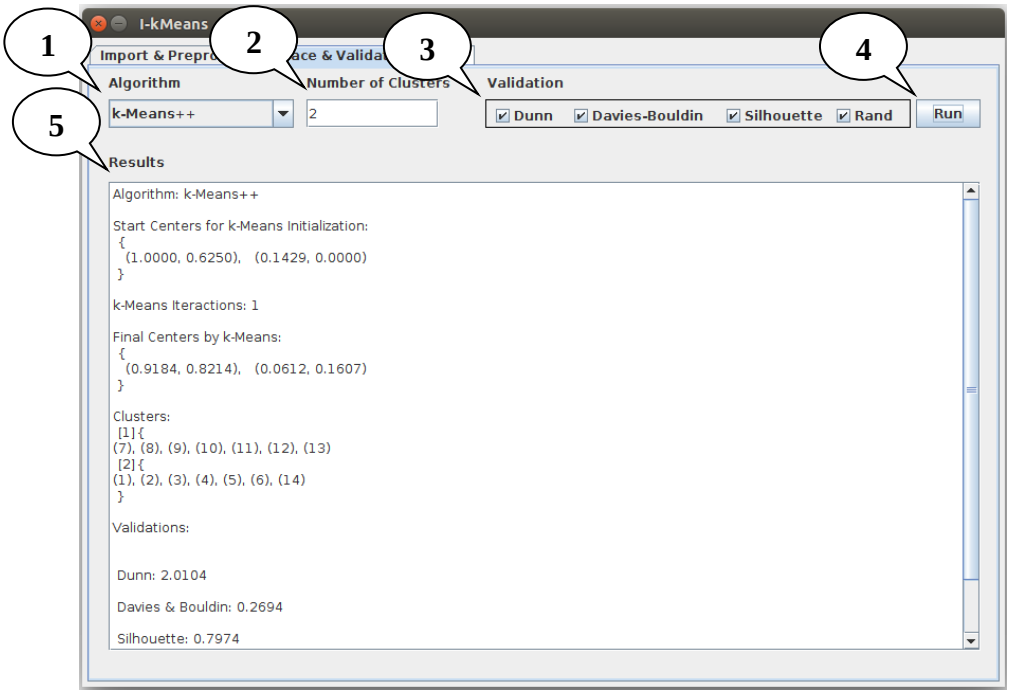


Figura 6.3 Painel "Trace & Validation" com marcação e enumeração.

(1) O campo "Algorithm" permite que o usuário escolha o algoritmo a ser executado, entre os cinco implementados *i.e.*, ++, SPSS, M1, CCIA e MS.

(2) O campo "Number of Clusters" permite que o usuário escolha o número de grupos que o algoritmo deve considerar, sendo 3 o valor *default* desse campo.

(3) O campo "Validation" permite que o usuário escolha qual(is) validação(ões) pode utilizar (dentre as disponibilizadas no painel). Caso não haja escolha, nenhuma é feita e caso algumas (ou todas) sejam escolhidas, o sistema informa os respectivos índices obtidos.

(4) O botão "Run" deve ser acionado apenas após a escolha do algoritmo a ser executado, da definição do número de grupo e de quais validações devem ser conduzidas. Quando o botão "Run" é acionado, o algoritmo escolhido é executado, o que faz com que o campo "Results" seja preenchido, com os resultados da execução e com os valores dos índices de validação escolhidos.

(5) O campo "Results" exibe os resultados obtidos pelo algoritmo. São informados o nome do algoritmo, os centroides iniciais encontrados pelo algoritmo, o número de iterações até a convergência do k-Means, os centroides finais após convergência, os grupos com as instâncias associadas a eles e, finalmente, os valores dos índices de validações escolhidos.

6.1.3 Módulo Plotagem no Plano Cartesiano (painel Plot)

O módulo de plotagem é o responsável pela exibição do conjunto de instâncias carregado em memória, no plano cartesiano, após o arquivo de dados ter sido carregado e antes da execução do algoritmo. O módulo permite a escolha, por parte do usuário, de dois atributos (dentre aqueles que descrevem as instâncias) a serem plotados. Na eventualidade das instâncias terem classes associadas, cada classe é identificada na plotagem do conjunto por meio de uma cor, diferente das demais.

Na Figura 6.4 as funcionalidades do módulo são mostradas por meio de marcação e enumeração de cada item do painel "Plot" do sistema I-kMeans. A plotagem inicial de um conjunto de instâncias definidas em um espaço bidimensional está mostrada na figura.

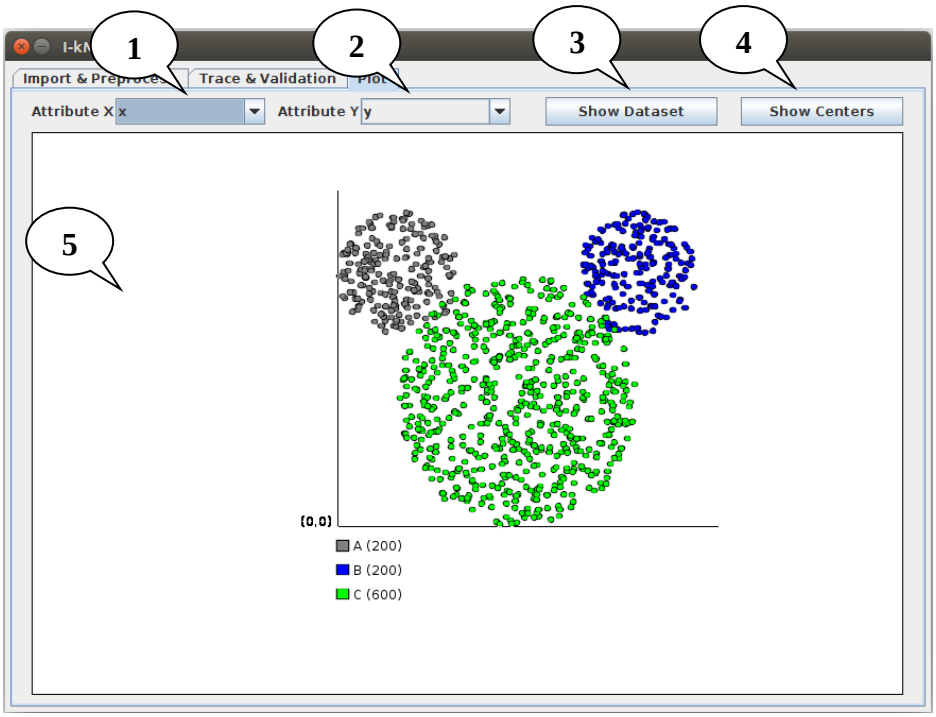


Figura 6.4 Painel "Plot" com marcação, enumeração e a plotagem do conjunto de instâncias antes da execução do algoritmo, cada cor (no caso 3) representa uma das classes informadas no conjunto original de instâncias fornecidas. O painel também apresenta uma legenda informando os nomes das classes e o número de instâncias por classe.

(1) O campo "Attribute x" permite a escolha do atributo a ser usado na coordenada x para plotagem no plano cartesiano.

(2) O campo "Attribute y" permite a escolha do atributo a ser usado na coordenada y para plotagem no plano cartesiano.

(3) O botão “Show Dataset” permite visualização da plotagem do conjunto original de instâncias de acordo com a seleção dos campos de coordenadas x e y.

(4) O botão “Show Centers” permite atualização da plotagem após a execução de um algoritmo pelo painel “Trace & Validation” de acordo com a seleção dos campos de coordenadas x e y.

(5) O plano cartesiano mostra a plotagem do conjunto de instâncias de acordo com a seleção dos campos de coordenadas x e y. Cada instância é plotada em cores, em que cada cor representa uma classe (caso essa informação exista), do conjunto original de instâncias. Na ausência da classe associada às instâncias, elas são plotadas em uma única cor cinza. Após a execução do algoritmo o usuário, ao atualizar a plotagem clicando no botão “Show Centers”, os centroides e agrupamentos finais são mostrados. Os centroides finais determinados pelo algoritmo são representados por pontos pretos com dimensão maior que a dos demais. Os centroides calculados pelo algoritmo, ao longo de sua execução, são representados por pontos vermelhos com dimensão maior que a dos demais. Os demais pontos representam cada instância e cada grupo pode ser identificado por meio das instâncias de mesma cor. A Figura 6.5 mostra a plotagem após a execução do algoritmo ++.

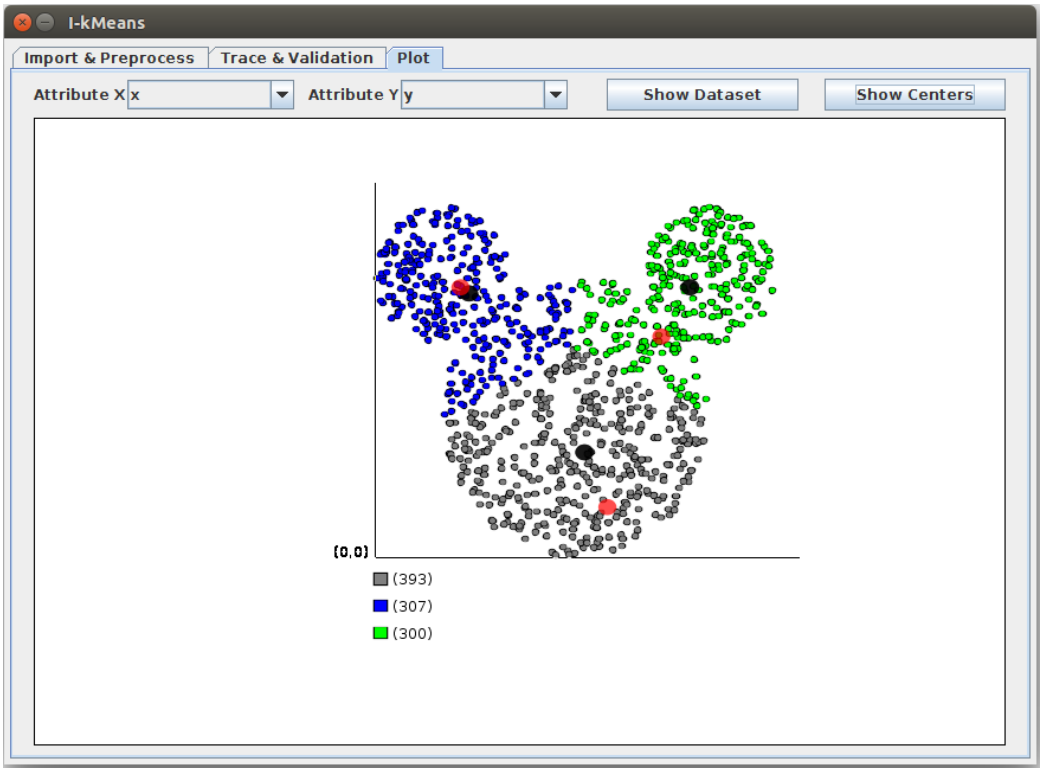


Figura 6.5 Painel "Plot" com a plotagem do conjunto de instâncias após a execução do algoritmo ++. Os centroides calculados pelo algoritmo escolhido estão representados por pontos vermelhos, após a convergência do algoritmo k-Means, representados por pontos pretos. Os três grupos do agrupamento obtido são identificados pelas cores distintas associadas às suas instâncias. O painel também apresenta a legenda com o número de instâncias por grupo.

6.2 Descrição dos Experimentos

Esta seção está organizada em três subseções. A Seção 6.2.1 apresenta os conjuntos de dados utilizados nos experimentos conduzidos, cujas características estão descritas na Tabela 6.1. A Seção 6.2.2 descreve a metodologia seguida para a realização dos experimentos. Com o objetivo de organizar a apresentação/discussão dos resultados dos experimentos, a Seção 6.2.3 está dividida em três partes; as duas primeiras que tratam dos experimentos relacionados a conjuntos de dados artificiais e conjuntos de dados reais, respectivamente e a terceira resume os resultados/discussões apresentados das duas primeiras.

6.2.1 Conjuntos de Dados Utilizados

Para avaliar os algoritmos implementados foram utilizados oito conjuntos de dados sintéticos e quatro conjuntos de dados reais em que três foram extraídos do UCI Repository [Dua & Karra Taniskidou 2017] e o outro encontrado em [Herman 1971].

Durante o desenvolvimento e implementação do I-kMeans, foram selecionados alguns conjuntos de dados para avaliar a eficácia de cada algoritmo. Para teste do algoritmo M1 foi necessário criar um conjunto de dados, visto que nos conjuntos escolhidos o algoritmo não se comportou como descrito em [Al-Daoud & Roberts 1996]. Infelizmente os autores do algoritmo, na publicação em que o descrevem, não citaram a fonte do conjuntos de dados por eles utilizado, o que não permite que as implementações do mesmo algoritmo possam ser comparadas.

Os conjuntos de dados sintéticos utilizados nos experimentos foram: Ruspini [Ruspini 1970], MSD [Maedeh & Suresh 2013] (para diferenciar os dados usados por Maedeh & Suresh, do algoritmo proposto por esses dois autores, nomeado de MD, decidiu-se nomear os dados de MSD), Mouse-Like [GMUM.r 2018], Spherical_6_2 [Bandyopadhyay & Maulik 2002], LongSquare [Handl *et al.* 2004], Aggregation [Gionis *et al.* 2007], 3MC [Su *et al.* 2005] e um conjunto de dados chamado Teste_M1, criado pelo autor dessa dissertação. A Tabela 6.1 apresenta as principais características dos oito conjuntos de dados bidimensionais e a Figura 6.6 mostra tais conjuntos plotados no plano cartesiano bidimensional, considerando que todos os oito conjuntos são conjuntos de dados bidimensionais.

Além dos conjuntos de dados bidimensionais apresentados na Tabela 6.1, foram também utilizados mais quatro conjuntos de dados reais, cujas características estão apresentadas na Tabela 6.2: Iris [Fisher 1936], Fossil [Herman 1971], Wine e Seeds disponíveis junto ao UCI Repository [Dua & Karra Taniskidou 2017].

Tabela 6.1 Características dos conjuntos de dados utilizados nos experimentos. #NI: número de instâncias, #NA: número de atributos, #NG: número de grupos e G = #NI: Número de instâncias por grupo, G_id: identificação do grupo.

Identificação	#NI	#NA	#NG	G_id = #NI
Ruspini	75	2	4	1=20, 2=23, 3=17, 4=15
MSD	30	2	5	1=4, 2=7, 3=10, 4=4, 5=5
Mouse-Like	1.000	2	3	1=200, 2=200, 3=600
Spherical_6_2	250	2	5	1=50, 2=50, 3=50, 4=50, 5=50
LongSquare	900	2	6	1=147, 2=155, 3=150 4=148, 5=150, 6=150
Aggregation	788	2	7	1=45, 2=170, 3=102, 4=273, 5=34, 6=130, 7=34
3MC	400	2	3	1 = 120, 2 = 170, 3 = 170
Teste_M1	14	2	2	1=7,2=7

Tabela 6.2 Características dos conjuntos de dados reais utilizados nos experimentos. #NI: número de instâncias, #NA: número de atributos, #NG: número de grupos e G = #NI: Número de instâncias por grupo, G_id: identificação do grupo.

Identificação	#NI	#NA	#NG	G_id = #NI
Iris	150	4	3	1=50, 2=50 , 3=50
Fossil	87	6	3	1=40, 2=34 , 3=13
Wine	178	13	3	1=59, 2=71, 3=48
Seeds	210	7	3	1=70, 2=70 , 3=70

O conjunto de dados Iris [Fisher 1936] é muito utilizado pela comunidade de AM e os dados relativos ao Iris foram baixados do UCI Repository [Dua & Karra Taniskidou 2017]. Esse conjunto de dados possui três classes representando três tipos da flor Iris, (I) Iris Setosa, (II) Iris Versicolor e (III) Iris Virginica. Cada classe contém 50 instâncias, totalizando 150 instâncias no conjunto de dados completo. Cada instância é descrita por quatro atributos, relativos às medidas (em cm) de quatro características da flor Iris, que são: *comprimento da sépala, largura da sépala, comprimento da pétala, largura da pétala*.

O conjunto de dados Fossil [Herman 1971], utilizado na avaliação do algoritmo CCIA, descrita em [Khan & Ahmad 2004], consiste de 87 instâncias de dados, cada uma delas representando uma amostra do que é identificado presentemente como fossil *nummulite*. Tais amostras foram extraídas da formação de calcário amarelo do noroeste da Jamaica e remontam ao período pré-histórico conhecido como eoceno. Tal período sucede o período paleoceno e precede o período oligoceno. Cada amostra é caracterizada por seis atributos. Como identificado por Chernoff em [Chernoff 1971], o conjunto de dados pode ser abordado dividido em três grupos, com os seguintes números de instâncias em cada um deles: (I) 40, (II) 34 e (III) 13.

O conjunto de dados Wine está disponível junto ao UCI Repository [Dua & Karra Taniskidou 2017] e possui 178 instâncias que são resultados da análise química de vinhos provenientes de uvas cultivadas em uma mesma região da Itália; os dados são derivados de três diferentes cultivos. A análise determinou treze atributos fundamentais que caracterizam os três grupos de vinhos. Os três grupos têm os seguintes números de instâncias em cada um deles: (I) 59, (II) 71 e (III) 48.

O Conjunto de dados Seeds está também disponível junto ao UCI Repository [Dua & Karra Taniskidou 2017] e consiste de um total de 210 instâncias, em que cada uma delas descreve um grão de trigo, de colheitas de trigo realizadas em campos experimentais do Instituto de Agrofísica da Academia Polaca de Ciências em Lublin.

Esse conjunto de dados possui instâncias de dados que descrevem três diferentes grupos de trigo: (1) Kama, (2) Rosa e (III) Canadense sendo que cada grupo possui 70 instâncias.

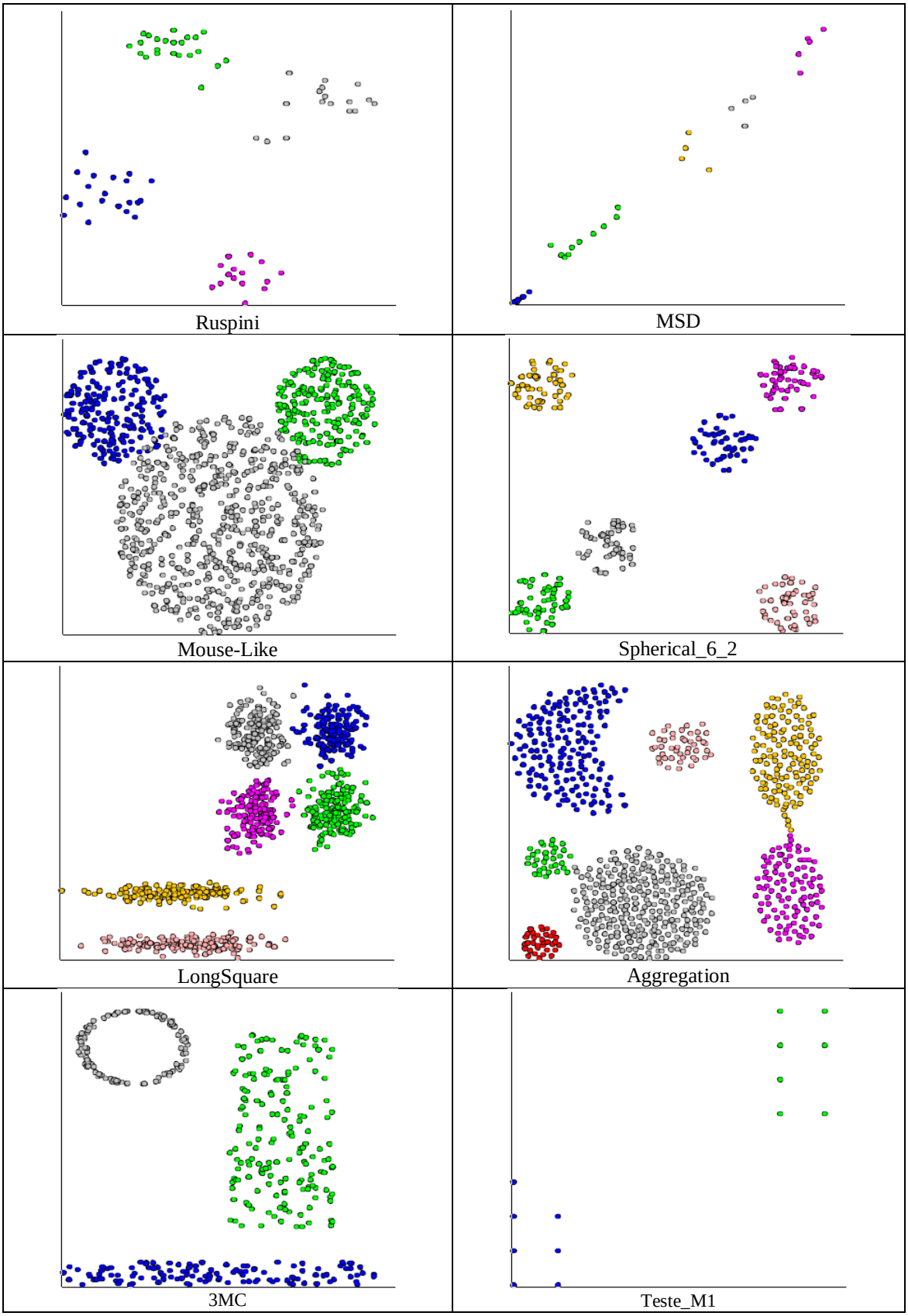


Figura 6.6 Visualização dos conjuntos de dados artificiais utilizados nos experimentos (Tabela 6.1).

6.2.2 Metodologia Adotada para a Realização dos Experimentos

A metodologia adotada para comparação dos desempenhos dos cinco algoritmos de inicialização do k-Means foi implementada por meio da execução sequencial dos passos descritos a seguir, para cada um dos conjuntos de dados (genericamente identificado como X) apresentado na Seção 6.2.1, e listados nas Tabela 6.1 e Tabela 6.2, considerando um dos algoritmos de inicialização Y (em que Y indica o algoritmo utilizado: k-Means, ++ ou SPSS ou CCIA ou M1 ou MS). Como será descrito oportunamente, o k-Means original, em que a seleção dos centroides iniciais é feita de maneira randômica, foi também executado no mesmo esquema que aquele associado aos algoritmos ++ e M1, com vistas às comparações com os resultados obtidos em que as escolhas iniciais de centroides tenham sido feitas via um dos cinco algoritmos investigados.

A metodologia discrimina os algoritmos em dois conjuntos, {k-Means, ++, M1} e {CCIA, SPSS, MS}, devido ao fato dos três algoritmos, k-Means, ++ e M1, envolverem uso de procedimento com escolhas randômicas e os outros três não. Essas escolhas ocorrem quando o k-Means original escolhe os k centroides de maneira randômica, o ++ escolhe o primeiro centroide e quando o M1 faz uma seleção randômica de instâncias dentro de uma célula e, também, ao final, quando determinadas condições não são satisfeitas.

Para {k-Means, ++ e M1}, decidiu-se que as etapas (2.1), (2.2) e (2.3) deveriam ser executadas 20 vezes; tal número foi escolhido por terem sidos utilizados em experimentos descritos em [Arthur & Vassilvitskii 2007] e em [Pavan *et al.* 2010], nos experimentos realizados por esses autores, utilizando o algoritmo ++.

A publicação [Al-Daoud & Roberts 1996] em que o algoritmo M1 é proposto, não tem qualquer informação relativa ao número de execuções realizadas nos experimentos conduzidos pelos autores; também deve ser considerado que os conjuntos de dados utilizados pelos autores foram bem específicos, o que evitou que o algoritmo M1 finalizasse o processamento com a escolha randômica de centroides (fato que ocorreu com uma certa frequência quando dos experimentos realizados neste trabalho, apresentados e discutidos nas Seção 6.2.3.1 e Seção 6.3.2.2). Para os algoritmos CCIA, SPSS e MS, que fornecem sempre os mesmos centroides para um determinado conjunto de dados quando executado mais de uma vez, apenas uma execução foi conduzida para

cada um dos três algoritmos. A metodologia de experimentação adotada, portanto, segue os passos descritos a seguir.

(1) Atribuir ao parâmetro k o número de grupos visivelmente identificáveis ou número de classes presentes no conjunto de dados considerado X;

(2) Para {k-Means, ++, M1} executar os passos (2.1), (2.2) e (2.3) vinte vezes (devido às escolhas randômicas envolvidas) e para os algoritmos {CCIA, SPSS, MS} uma vez, dado que nenhum dos três algoritmos faz uso de escolhas randômicas.

(2.1) Executar o algoritmo de inicialização Y, usando o conjunto X, cujo resultado de execução é um conjunto de k centroides C;

(2.2) Executar o k-Means sem seu passo de inicialização original, usando como entrada, além do conjunto X, o conjunto de k centroides C criado no passo (2.1), obtendo o agrupamento AG;

(2.3) Calcular os índices de validação Rand, Sillhouette, Davies-Bouldin e Dunn no agrupamento AG obtido em (2.2) e armazená-los;

(3) Para {k-Means, ++ e M1} calcular a média e desvio-padrão dos valores de validação e do número de iterações realizadas pelo k-Means, para convergir, nas 20 execuções realizadas nos passos (2.1), (2.2) e (2.3). Para os demais algoritmos CCIA, SPSS e MS os valores encontrados em uma única execução dos passos (2.1), (2.2) e (2.3) serão utilizados, considerando que esses algoritmos apresentam sempre o mesmo resultado quando executados mais de uma vez tendo o mesmo conjunto de dados como entrada.

6.2.3 Apresentação dos Resultados Obtidos e Discussão

Esta subseção foi dividida em três outras subseções, com o objetivo de agrupar os experimentos com relação ao tipo de dado utilizado ou seja, a Subseção 6.2.3.1 diz respeito aos dados artificiais da Tabela 6.1, plotados na Figura 6.6. A Subseção 6.2.3.2 aborda os experimentos realizados considerando os quatro conjuntos de dados reais, que foram baixados do UCI Repository [Dua & Karra Taniskidou 2017], cujas

características estão apresentadas na Tabela 6.2; a terceira e última, Subseção 6.2.3.3, discute os resultados obtidos.

Devido à maneira como os algoritmos de inicialização se comportam (a) variam a inicialização a cada execução ou (b) não variam a inicialização a cada execução, para cada um dos doze domínios de dados, os resultados dos experimentos estão agrupados pela característica dos algoritmos de inicialização contemplar (a) ou (b). Os algoritmos que contemplam (a) são o k-Means, ++ e M1, enquanto que os que contemplam (b) são o CCIA, SPSS e MS. Por essa razão, os resultados dos dois grupos de algoritmos são mostrados, por conjunto de dados, divididos em duas tabelas.

Como o objetivo do trabalho foi o da investigação da real contribuição de métodos de inicialização para o k-Means, e como o k-Means tem o seu próprio método de inicialização (randômico), os resultados obtidos pelo k-Means original foram usados para comparações com aqueles obtidos pelo k-Means utilizando como inicialização os resultados de cada um dos cinco algoritmos investigados.

6.2.3.1 Resultados dos Experimentos Realizados com Oito Conjuntos de Dados Artificiais

Com o objetivo de organizar essa subseção e considerando o número de conjuntos de dados artificiais utilizados (8), os resultados dos experimentos são apresentados com foco em cada um dos conjuntos de dados utilizados, na ordem: Ruspini, MS, Mouse-Like, Spherical_6_2, LongSquare, Aggregation, 3MC e Teste_MI.

(A.1) Ruspini

A Tabela 6.3 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados identificados como Ruspini.

Tabela 6.3 (Ruspini) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,4894	0,1061	0,4225	0,1433	0,7159	0,0762	0,9843	0,0470	1,7000	1,0535
M1	0,4018	0,2064	0,5025	0,2246	0,6782	0,1095	0,9587	0,0715	2,8500	1,0618
k-Means	0,3110	0,2334	0,5729	0,2222	0,6313	0,1218	0,9292	0,0787	2,6500	1,1521

O algoritmo k-Means após receber como entrada os centroides encontrados pelo algoritmo ++ induziu agrupamentos com o maior valor de índice Rand associado e

atingiu a convergência com o menor número de iterações (em comparação com o seu desempenho quando inicializado com os outros dois métodos), como mostrado na Tabela 6.3.

Os valores do índice de Rand, associados aos agrupamentos induzidos utilizando inicializações fornecidas pelo MI e randômica (relativa à do próprio k-Means) também foram bons *i.e.*, considerando que tiveram valores do índice próximos de 1,0, e número de iterações realizadas pelo k-Means, inicializado com os centroides obtidos por esses dois algoritmos, em média, 2 vezes maior que o associado ao ++, muito embora ainda pequeno. Com relação aos índices D, DB e S associados aos agrupamentos obtidos pelo k-Means utilizando três diferentes inicializações, os valores são razoavelmente próximos, muito embora, novamente, os números sugerem que os agrupamentos criados pelo ++ sejam melhores.

A Tabela 6.4 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados identificados como Ruspini. A tabela exibe também o número de iterações realizadas pelo k-Means para atingir convergência, quando inicializado com os centroides fornecidos pelo CCIA, SPSS e MS, respectivamente.

Tabela 6.4 (Ruspini) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,521	0,3754	0,7413	1	0
SPSS	0,521	0,3754	0,7413	1	2
MS	0,521	0,3754	0,7413	1	1

Uma análise dos valores mostrados da Tabela 6.4, particularmente aqueles associados aos índices de validação, sugere que o mesmo agrupamento foi obtido pelo k-Means inicializado por qualquer dos três algoritmos de inicialização. Isso, de certa forma, implica que os centroides obtidos pelos três algoritmos de inicialização foram ou iguais ou quase iguais. Essa conclusão pode também ser corroborada pelo número de iterações realizadas pelo k-Means, para convergência.

O centroides iniciais fornecidos pelo CCIA foram centroides que não necessitaram ser 'corrigidos' pelo k-Means, uma vez que já satisfizeram o critério de convergência do k-Means. Já aqueles fornecidos pelo SPSS precisaram ligeira correção, conseguida via duas iterações do k-Means e aqueles fornecidos pelo MS, de apenas uma para o k-Means satisfazer o critério de convergência.

Os resultados obtidos neste experimento evidenciam que o ++ e todos os algoritmos que não fazem escolhas randômicas, forneceram inicializações ao k-Means que muito colaboraram para sua convergência rápida e, também, para o mesmo agrupamento que, particularmente quando do uso dos algoritmos CCIA, SPSS e MS, avaliado pelo S com uma qualidade entre razoável e boa.

(A.2) MSD

A Tabela 6.5 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados MSD.

Tabela 6.5 (MSD) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++, MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,4245	0,2301	0,6845	0,0082	0,9586	0,0457	0,9586	0,0457	2,4000	0,4245
M1	0,4018	0,2064	0,5025	0,2246	0,6782	0,1095	0,9587	0,0715	2,8500	0,4018
k-Means	0,2579	0,2498	0,4841	0,1311	0,6041	0,0958	0,9302	0,0642	4,5000	2,1563

Os agrupamentos obtidos com inicializações fornecidas pelos três algoritmos de inicialização tiveram, como média dos valores de índice R, valores próximos entre si e próximos de 1. Isso de certa forma implica que os respectivos agrupamentos obtidos foram bem próximos daqueles visualmente detectados. Considerando as médias dos valores do índice S, sem dúvida alguma o ++ forneceu inicializações que promoveram o k-Means a induzir agrupamentos melhores que as inicializações fornecidas pelo M1 e pela randômica. As inicializações fornecidas pelo ++ também promoveram um menor número de iterações próximo da metade, do k-Means, em média, para atingir convergência.

A Tabela 6.6 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados MSD.

Tabela 6.6 (MSD) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,014	0,5018	0,3403	0,8391	7
SPSS	0,1701	0,4507	0,6754	0,908	1
MS	0,6328	0,4333	0,6920	1	2

Analisando os valores do índice R mostrados na Tabela 6.6 pode ser constatado que os agrupamentos induzidos pelo k-Means, utilizando como centroides iniciais aqueles encontrados pelo CCIA, SPSS e pelo MS, foram de boa qualidade.

Entretanto, como o conjunto de dados MSD foi o conjunto de dados proposto e utilizado pelos autores do algoritmo MS e, particularmente, aquele no qual o agrupamento induzido pelo k-Means, usando a inicialização fornecida pelo MS, teve o melhor valor de índice R, existe uma grande possibilidade do conjunto MSD ser um conjunto construído de maneira conveniente ao algoritmo MS. Tal possibilidade foi conjecturada tendo em conta o conhecimento adquirido sobre o algoritmo quando de seu estudo e implementação, durante a realização deste trabalho. O agrupamento induzido pela inicialização fornecida pelo MS também obteve o melhor dos três valores do índice D, índice DB e de índice S.

(A.3) Mouse-Like

A Tabela 6.7 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Mouse-Like.

Tabela 6.7 (Mouse-Like) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,0194	0,0015	0,6865	0,0005	0,5041	0,0003	0,7339	0,0017	6,8000	2,9597
M1	0,0191	0,0014	0,6865	0,0007	0,5040	0,0002	0,7334	0,0012	7,7500	2,1650
k-Means	0,0197	0,0016	0,6868	0,0008	0,5041	0,0003	0,7341	0,0016	7,2500	2,2555

Por meio da análise dos valores da média dos quatro índices de validação utilizados mostrados na Tabela 6.7 pode ser conjecturado que, particularmente no domínio de dados Mouse-Like, os três métodos de inicialização de centroides utilizados induziram o k-Means a produzir agrupamentos bastante similares entre si. Também, tal similaridade entre eles se reflete, mas não tão acentuadamente, na média do número de iterações conduzidas pelo k-Means, para atingir convergência.

A Tabela 6.8 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Mouse-Like.

Tabela 6.8 (Mouse-Like) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,0183	0,6862	0,5039	0,7328	5
SPSS	0,0216	0,6872	0,5047	0,7368	6
MS	0,0216	0,6872	0,5047	0,7368	5

O mesmo padrão de comportamento de valores de índices de validação de agrupamentos, obtidos com os algoritmos k-Means, ++ e M1 mostrados na Tabela 6.7 pode ser observado nos valores de tais índices associados aos agrupamentos obtidos pelo CCIA, SPSS e MS, mostrados na Tabela 6.8.

Os valores de DB, S e R são praticamente os mesmos e, considerando que o número de iterações realizadas pelo k-Means, inicializado com os centroides fornecidos por CCIA, SPSS e MS ser quase o mesmo, pode ser inferido que os três agrupamentos induzidos, cada um a partir de centroides gerados por um dos três algoritmos sendo considerados, é praticamente o mesmo.

(A.4) Spherical_6_2

A Tabela 6.9 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Spherical.

Tabela 6.9 (Spherical) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means, pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,3028	0,2344	0,7244	0,3263	0,6894	0,0661	0,9686	0,0346	3,1000	2,7730
M1	0,5149	0,0000	0,4369	0,0000	0,7481	0,0000	1,0000	0,0000	2,3500	0,7262
k-Means	0,1240	0,1693	0,9229	0,2588	0,6188	0,0645	0,9291	0,0389	5,1500	2,7730

O valor médio do índice R associado ao agrupamento obtido pelo k-Means, com inicialização fornecida pelo M1 foi ótimo, M1 também obtém melhores valores dos índices D, DB e S comparados ao ++ e k-Means quando inicializado randomicamente.

Como o número de iterações realizadas pelo k-Means até atingir convergência foi, em média, ligeiramente diferente, M1 obtém menor número de iteração dos três valores mostrados na tabela considerando as três diferentes inicializações, pode ser conjecturado que as inicializações fornecidas foram diferenciadas, mas não muito, a ponto de exigir do k-Means um número maior de iterações para atingir convergência.

É importante mencionar que o algoritmo M1 induziu o mesmo agrupamento, nas 20 vezes que o processo foi repetido, cuja inicialização colaborou na indução, pelo k-Means, de agrupamentos que, em média, tiveram valor = 1,0 associado ao índice R, tais agrupamentos também tiveram, em média, o melhor valor associado do índice DB. O valor médio dos índices D associados aos agrupamentos obtidos não foi bom e, aqueles associados à agrupamentos obtidos com a inicialização randômica, foram os piores.

A Tabela 6.10 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Spherical.

Tabela 6.10 (Spherical) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,5149	0,4369	0,7481	1,00	0
SPSS	0,0204	1,4689	0,5961	0,9305	5
MS	0,5149	0,4369	0,7481	1,00	1

De maneira semelhante aos resultados obtidos com o k-Means, inicializado com os algoritmos ++, M1 e de maneira randômica, os resultados apresentados na Tabela 6.10, relativos aos desempenhos de valores de inicialização fornecidos pelo CCIA, SPSS e MS no mesmo domínio de dados também foram similares entre si. Particularmente, o valor do índice R associado aos três agrupamentos induzidos pelo k-Means, utilizando as inicializações obtidas pelo CCIA, SPSS e MS, são indicativos que os agrupamentos induzidos são semelhantes àquele detectado visualmente.

(A.5) LongSquare

A Tabela 6.11 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Long Square.

Tabela 6.11 (LongSquare) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,0202	0,0062	0,9962	0,1114	0,5162	0,0237	0,9367	0,0290	8,9000	4,1821
M1	0,0203	0,0057	0,9983	0,0947	0,5153	0,0237	0,9338	0,0259	12,4500	4,7061
k-Means	0,0152	0,0064	0,9658	0,1538	0,4974	0,0216	0,9240	0,0205	12,3000	6,1081

Os agrupamentos induzidos pelo k-Means usando as inicializações obtidas pelo ++, M1 e feitas de maneira randômica, tiveram a média de valores associados aos

índices DB de quase 1, o que é um indicativo que, no domínio de dados LongSquare, os grupos não são tão compactos e distantes uns dos outros.

Também, a média dos valores do índice D associados aos agrupamentos obtidos é bem baixa, o que pode ser considerado um indicativo que os agrupamentos obtidos não refletem muito a natureza dos dados. No entanto, a média dos valores do índice R associados aos agrupamentos obtidos pelo k-Means, inicializado por ++, M1 e randomicamente, é próxima de 1, indicativo de agrupamentos que refletem o agrupamento visualmente identificado.

Analisando os valores do índice S pode ser observado que as médias de tal índice obtidas dos agrupamentos induzidos, estão por volta de 0,5, o que aponta para agrupamentos razoáveis, mas não tão bons, considerando que os valores de S variam ao intervalo [–1 1]. Como é um domínio de dados que contém 6 grupos, com aproximadamente 150 instâncias/grupo, o número médio de iterações necessárias para o k-Means convergir, quando inicializado randomicamente ou via M1, ficou por volta de 12 iterações, enquanto que quando iniciado pelo ++, necessitou em média, aproximadamente 9 iterações para atingir convergência.

Esse é um domínio de dados em que, particularmente, os índices de validação dos agrupamentos obtidos estão ligeiramente desfocados. Enquanto o índice R aponta para bons agrupamentos, embora com um número maior de iterações, os índices D, DB indicam agrupamentos não muito bons e o índice S agrupamentos razoáveis.

Tabela 6.12 (LongSquare) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,023	0,947	0,4645	0,8602	30
SPSS	0,0242	1,0543	0,5309	0,9445	2
MS	0,0242	1,0543	0,5309	0,9445	4

A mesma tendência observada quando da análise de valores de índices de agrupamentos gerados pelo k-Means, com inicializações pelo ++, M1 e randômica, mostradas na Tabela 6.11, pode ser observada nos valores mostrados na Tabela 6.12, associados aos algoritmos CCIA, SPSS e MS.

O fato do k-Means ter realizado 30 iterações para atingir convergência é um indicativo que a forma como o CCIA trabalha (*i.e.*, a própria concepção do algoritmo, que é razoavelmente elaborada e trabalhosa), não é adequada para a identificação de bons centroides iniciais no domínio LongSquare. O CCIA produziu uma inicialização

não tão bem posicionada, o que provocou a necessidade do k-Means realizar 30 iterações para convergir, obtendo um agrupamento que, de acordo com o índice R, ficou aquém dos agrupamentos induzidos pelo k-Means quando inicializado pelo SPSS e MS.

Os valores dos índices D, DB, S e R para os agrupamentos inicializados pelo SPSS e MS são iguais, o que é um indicativo que o k-Means induziu o mesmo agrupamento, com as inicializações fornecidas pelo SPSS e pelo MS, sendo que necessitou de duas iterações a mais para convergir, quando a inicialização foi fornecida pelo MS.

(A.6) Aggregation

A Tabela 6.13 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Aggretation.

Tabela 6.13 (Aggregation) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,0294	0,0048	1,0017	0,0759	0,4716	0,0255	0,9099	0,0158	13,8500	3,6094
M1	0,0292	0,0062	0,9775	0,0855	0,4644	0,0301	0,9030	0,0157	15,5000	5,7576
k-Means	0,0314	0,0057	0,9688	0,0845	0,4582	0,0175	0,8898	0,0089	18,0000	6,8920

O Aggregation é um domínio de dados que pode apresentar problemas para algoritmos de agrupamentos e, também, algoritmos de inicialização de centroides. No Aggregation podem ser visualmente detectados 7 grupos. Por um lado os valores apresentados na Tabela 6.13, relativos às médias de valores dos índices D, DB e S, sugerem que os agrupamentos obtidos pelo k-Means, inicializado pelo ++, M1 e randomicamente, não foram muito bons. Por outro lado, entretanto, os valores associados ao índice R evidenciam a indução de agrupamentos similares àqueles visualmente detectados.

A configuração e disposição dos dados no Aggregation não é particularmente adequada à uma abordagem baseada em grid, como é aquela adotada pelo M1. O Algoritmo M1 não foi capaz de encontrar centroides por meio de sua abordagem baseada em grid, o que fez com que, em tais situações, adotasse escolhas aleatórias de centroides, que é o mesmo procedimento adotado na concepção do k-Means original. Particularmente, no domínio de dados Aggregation, a atuação do M1 é exatamente a mesma do k-Means original, em que centroides iniciais são escolhidos randomicamente.

A Tabela 6.14 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Aggretation.

Tabela 6.14 (Aggregation) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,0339	0,9881	0,4843	0,9202	20
SPSS	0,0340	0,9888	0,4919	0,9110	13
MS	0.0319	0.9023	0.4047	0.8830	28

A mesma tendência observada nos dados apresentados na Tabela 6.13 é observada nos dados apresentados na Tabela 6.14. Por um lado os valores apresentados na Tabela 6.14, relativos aos valores dos índices D, DB e S, sugerem que os agrupamentos obtidos pelo k-Means, inicializado pelo CCIA, SPSS e MS, não foram muito bons. Por outro lado, entretanto, os valores associados ao índice R evidenciam a indução de agrupamentos similares àqueles visualmente detectados. Embora os números de iterações realizadas pelo k-Means, considerando os centroides iniciais que lhe são passados, tenham sido altos, eles, de certa forma, acompanham a tendência do número médio de iterações feitas pelo k-Means, quando do uso de centroides iniciais calculados pelo ++, M1 e randômico.

Revendo os dados mostrados nas tabelas 6.13 e 6.14, que englobam os cinco algoritmos de inicialização *i.e.*, ++, SPSS, CCIA, M1 e MS, e comparando-os com os dados exibidos na Tabela 6.13, relativos ao algoritmo k-Means, como originalmente proposto [MacQueen 1997], pode-se afirmar que os centroides iniciais fornecidos pelos dois algoritmos, CCIA e MS, não conseguiram minimizar o número de iterações necessárias para que o algoritmo k-Means atingisse a convergência. Essa constatação evidencia que, no domínio de dados Aggregation, os dois algoritmos de inicialização foram ineficientes para a tarefa. Já o algoritmo SPSS apresentou bons resultados fazendo que o algoritmo k-Means atingisse convergência com 13 iterações e produzisse um agrupamento com valor do índice R de 91,10%.

(A.7) 3MC

A Tabela 6.15 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados 3MC.

Tabela 6.15 (3MC) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,0224	0,0019	0,7885	0,0273	0,5105	0,0051	0,8970	0,0618	6,8500	2,6509
M1	0,0234	0,0027	0,7699	0,0368	0,5066	0,0074	0,8541	0,0843	7,8	3,9949
k-Means	0,0239	0,0028	0,7623	0,7623	0,5051	0,0074	0,8367	0,0863	6,9500	2,6921

Os dados mostrados na Tabela 6.15, relativos às médias de valores do índice de validação R, indicam que a qualidade dos agrupamentos induzidos no domínio 3MC pode ser assumida como boa, ou seja, os agrupamentos induzidos se aproximam daqueles que podem ser detectados visualmente no domínio 3MC. As médias de valores de R, entretanto, privilegiam os agrupamentos induzidos pelo k-Means quando inicializado pelo ++, M1 e randomicamente, nesta ordem. Os valores dos outros índices, D e DB não caracterizam os agrupamentos obtidos com bons e os valores de S o qualificam como um pouco acima de razoável.

Como aconteceu no domínio Aggregation, novamente o algoritmo M1 não conseguiu encontrar centroides de acordo com a abordagem grid que esse algoritmo adota, o que fez com que o algoritmo adotasse a sua solução para tais situações, que é a da escolha randômica de centroides, o que faz com que o M1 se comporte exatamente como o k-Means original.

A Tabela 6.16 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados 3MC.

Tabela 6.16 (3MC) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,0216	0,8000	0,5126	0,9230	0
SPSS	0,0270	0,7235	0,4983	0,7498	8
MS	0,0270	0,7235	0,4983	0,7498	7

Uma análise dos dados mostrados na Tabela 6.16 evidencia que o CCIA induziu uma inicialização de centroides para o k-Means que não exigiu do k-Means qualquer iteração para atingir a convergência. O conjunto de centroides iniciais fornecido pelo CCIA já satisfez o critério de convergência do k-Means e, portanto, o número de iterações necessárias foi zero, como indicado na tabela. O valor do índice R associado ao agrupamento obtido foi o melhor dos três valores mostrados na tabela, um indicativo que o CCIA colaborou com o k-Means a ponto do k-Means não ter que realizar

qualquer trabalho e, também, obteve um conjunto de centroides cujo agrupamento associado é bem próximo daquele visualmente detectado.

Ao que tudo indica os centroides iniciais obtidos pelo SPSS e MS eram bem próximos, uma vez que o k-Means necessitou de 8 e 7 iterações, respectivamente, para induzir o mesmo agrupamento; o fato de ser o mesmo agrupamento pode ser deduzido com base nos valores dos índices D, DB, S e R associados serem os mesmos.

(A.8) Teste_MI

A Tabela 6.17 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Teste_MI.

Tabela 6.17 (Teste_MI) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	2,0104	0,0000	0,2694	0,0000	0,7974	0,0000	1,0000	0,0000	1,1000	0,3000
M1	2,0104	0,0000	0,2694	0,0000	0,7974	0,0000	1,0000	0,0000	1,0000	0,0000
k-Means	2,0104	0,0000	0,2694	0,0000	0,7974	0,0000	1,0000	0,0000	1,4000	0,4898

O conjunto de dados Teste_MI foi criado como um conjunto de dados de controle, durante o desenvolvimento do sistema computacional que disponibiliza as implementações dos algoritmos de inicialização de centroides, com o objetivo específico de acompanhar e avaliar o funcionamento de algoritmos de inicialização baseados em grid, no caso, o M1. É um conjunto de dados bem simples, com dois grupos bem separadas, cada um com 7 instâncias.

Como evidenciam os dados mostrados na Tabela 6.17, o k-Means, utilizando as inicializações produzidas pelo ++, M1 e de maneira randômica, induziu o mesmo agrupamento, nas 20 vezes que o processo foi repetido, considerando que as médias dos valores dos índices associados aos agrupamentos obtidos foram as mesmas. Entretanto, houve uma pequena variação no número de iterações necessárias ao k-Means, para atingir convergência, o que pode ser notado na pequena variação na média do número de iterações realizadas pelo k-Means.

A Tabela 6.18 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Teste_MI.

Tabela 6.18 (Teste_MI) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	2,0104	0,2694	0,7974	1	0
SPSS	2,0104	0,2694	0,7974	1	1
MS	2,0104	0,2694	0,7974	1	1

A mesma tendência observada com os dados mostrados na Tabela 6.17, pode ser observada nos dados da Tabela 6.18. O fato do número de iterações do k-Means ter sido 0, quando usa como centroides iniciais aqueles fornecidos pelo CCIA é evidência que o k-Means já satisfaz o critério de convergência com os centroides iniciais fornecidos, não necessitando realizar qualquer processamento de ajuste dos centroides.

O fato dos quatro índices de validação terem o mesmo valor é evidência que os três agrupamentos induzidos pelo k-Means utilizando os centroides que lhe foram passados pelo CCIA, SPSS e MS resultou em três agrupamentos iguais.

6.2.3.2 Resultados dos Experimentos Realizados com Quatro Conjuntos de Dados Reais (UCI Repository)

Com o objetivo de organizar essa subseção e considerando os 4 conjuntos de dados reais utilizados, os resultados dos experimentos são apresentados com foco em cada um dos conjuntos de dados, na ordem: Iris, Fossil, Wine e Seeds.

(B.1) Iris

A Tabela 6.19 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Iris.

Tabela 6.19 (Iris) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,0601	0,0085	0,9352	0,0358	0,4879	0,0216	0,8556	0,0454	5,4000	1,4966
M1	0,0564	0,0077	0,9514	0,0407	0,4783	0,0255	0,8397	0,0600	5,5000	2,2693
k-Means	0,0609	0,0085	0,9331	0,0363	0,4889	0,0218	0,8559	0,0455	6,2500	2,4469

Todos os resultados apresentados na Tabela 6.19, tanto os relativos às médias dos valores dos índices de validação, quanto os relativos às médias do número de iterações

executadas pelo k-Means, com centroides inicializados utilizando o ++, M1 e escolha randômica, não são estatisticamente muito diferentes entre si. No domínio Iris, o resultado ligeiramente melhor na média, do número de iterações realizada pelo k-Means, quando as inicializações foram fornecidas pelo ++ e M1 talvez nem compense o uso de inicializadores diferentes da escolha randômica inicial que o k-Means original implementa.

A Tabela 6.20 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Iris.

Tabela 6.20 (Iris) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,069	0,9046	0,5048	0,8737	3
SPSS	0,053	0,9458	0,4838	0,8679	7
MS	0,053	0,9458	0,4838	0,8679	9

Com exceção do número de iterações, todos os outros valores apresentados na Tabela 6.20 seguem a mesma tendência daqueles apresentados na Tabela 6.19, ou seja, não diferem entre si de maneira significativa. Pode ser observado na Tabela 6.20, entretanto, que o algoritmo CCIA forneceu ao k-Means o melhor conjunto de 3 centroides iniciais, considerando que o k-Means necessitou apenas de 3 iterações para atingir convergência, enquanto que com centroides iniciais fornecidos pelo SPSS e pelo MS, o k-Means necessitou de realizar 7 e 9 iterações, respectivamente, para atingir convergência. No domínio Iris, particularmente, os métodos de inicialização SPSS e MS não colaboraram com o k-Means original, considerando que o k-Means original com a inicialização randômica, como mostrado na Tabela 6.19, teve, em média, que realizar 6,25 iterações para convergir.

(B.2) Fossil

A Tabela 6.21 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Fossil.

Tabela 6.21 (Fossil) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,3433	0,0856	0,9827	0,2094	0,4664	0,0731	0,9619	0,0784	5,3000	3,3926
M1	0,2570	0,1070	1,2182	0,3139	0,3902	0,0987	0,8900	0,0955	4,1000	1,7578
k-Means	0,2927	0,1153	1,1601	0,4205	0,4325	0,0903	0,9318	0,0880	4,7500	2,6433

Analisando os dados mostrados na Tabela 6.21 nota-se que, com relação à média de valores do índice R relativos aos agrupamentos obtidos, a inicialização fornecida pelo ++, na média, ligeiramente favoreceu a qualidade dos agrupamentos obtidos. Entretanto, em contrapartida, a média do número de iterações necessárias para o k-Means convergir foi ligeiramente mais alta quando comparada com a média de iterações necessárias ao k-Means, quando inicializado pelo M1 ou, então, pela escolha randômica, que é o método *default* utilizado pelo próprio k-Means. Com relação aos valores médios dos outros três índices, D, DB, e S, pode-se dizer que a qualidade dos agrupamentos obtidos, em geral, foi a mesma.

Como aconteceu anteriormente, o M1 não foi bem sucedido quando do uso de sua abordagem baseada em grid, o que fez com que o algoritmo, em todas as 20 execuções realizadas, tenha finalizado a busca por centroides iniciais por meio de uma escolha randômica, o que fez com que, no domínio Fossil, o M1 tenha se comportado exatamente como o k-Means original.

A Tabela 6.22 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Fossil.

Tabela 6.22 (Fossil) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,386	0,8784	0,5022	1	3
SPSS	0,133	1,3302	0,3839	0,8851	5
MS	0,386	0,8784	0,5022	1	2

Analisando os valores mostrados na Tabela 6.22, relativos ao índice R associado aos dois agrupamentos induzidos pelo k-Means, utilizando inicializações fornecidas pelo CCIA e MS, pode-se dizer que ambos agrupamentos foram bons e que ambos os algoritmos colaboraram para a diminuição do número de iterações necessárias ao k-Means para atingir a convergência e, de certa forma, ajudaram a promover a qualidade do agrupamento final obtido que, de acordo com o valor de R, foi o melhor possível.

O valor do índice R associado ao agrupamento obtido pelo k-Means com inicialização fornecida pelo SPSS, entretanto, apesar de ser um indicativo de um agrupamento relativamente bom, a inicialização considerada não contribuiu para um melhoramento do desempenho do k-Means original (Tabela 6.21), considerando que o k-Means, em média, necessitou de aproximadamente 5 interações para convergir para

agrupamentos cujos respectivos índices R foram melhores. O MS teve um bom desempenho e contribuiu com o k-Means fornecendo centroides iniciais que, em duas iterações do k-Means, induziram um agrupamento que reflete a organização em grupos que visualmente pode ser detectada nos dados do conjunto Fossil.

(B.3) Wine

A Tabela 6.23 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Wine.

Tabela 6.23 (Wine) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,1444	0,0153	1,3432	0,0104	0,3030	0,0015	0,9342	0,0083	4,8500	1,7684
M1	0,1424	0,0112	1,3454	0,0099	0,3030	0,0011	0,9346	0,0062	5,6000	2,3958
k-Means	0,1387	0,0036	1,3404	0,0106	0,3024	0,0005	0,9313	0,0030	4,7000	1,6462

Com relação aos valores médios do índice R associados aos agrupamentos obtidos com inicializações fornecidas pelos algoritmos ++, M1 e randômico, pode se dizer que todos os agrupamentos se aproximaram, a uma diferença de 0,007, do agrupamento formado pelos grupos de instâncias que compartilham a mesma classe, no domínio de dados Wine. Entretanto, nenhum dos dois métodos de inicialização, ++ e M1, colaboraram para promover um desempenho do k-Means, em número de iterações, que fosse melhor do que aquele obtido pela inicialização randômica utilizada pelo próprio k-Means.

Nos dados do domínio Wine o M1, novamente, não foi bem sucedido na busca por um conjunto inicial de centroides, por meio do método que implementa *i.e.*, baseado em grid e, então, recorreu à escolha randômica de tais centroides o que, como comentado anteriormente em situações em que tal problema ocorreu, faz com que o M1 se comporte como o k_Means original. Os valores médios dos índices D, DB e S associados aos agrupamentos obtidos indicam que tais agrupamentos são praticamente os mesmos, independentemente do método de inicialização empregado, nas vinte execuções do procedimento metodológico realizado para a obtenção dos resultados.

A Tabela 6.24 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Wine.

Tabela 6.24 (Wine) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D(↑)	DB (↓)	S(↑)	R(↑)	I
CCIA	0,142	1,3537	0,3025	0,9318	10
SPSS	0,135	1,3320	0,3028	0,9349	6
MS	0,135	1,3320	0,3028	0,9349	5

A mesma situação de valores de índices serem praticamente os mesmos, identificada nos valores apresentados na Tabela 6.23, é repetida nos valores mostrados na Tabela 6.24.

Por meio de uma análise com foco no número de iterações realizadas pelo k-Means para atingir convergência, pode ser constatado que o algoritmo CCIA teve um péssimo desempenho em relação ao conjunto de centroides iniciais que forneceu ao k-Means, uma vez que o k-Means teve que realizar 10 iterações para atingir convergência. De positivo, entretanto, pode se dizer que o investimento em tempo de processamento para convergir resultou na indução de um agrupamento, pelo k-Means, com um bom valor de índice R.

Levando em consideração os resultados apresentados em ambas, Tabela 6.23 e Tabela 6.24, pode ser afirmado que nenhum dos cinco algoritmos de inicialização foi capaz de obter um resultado que pudesse ser considerado o melhor, em termos de agrupamento obtido e número de iterações que o k-Means teve que realizar para chegar à convergência, do que a inicialização randômica, parte integrante do próprio k-Means original.

(B.4) Seeds

A Tabela 6.25 apresenta os resultados obtidos pelos algoritmos ++, M1 e k-Means utilizando os dados do domínio de dados Seeds.

Tabela 6.25 (Seeds) Resultados obtidos na execução dos algoritmos ++, M1 e k-Means pelo I-kMeans com média (μ) e desvio padrão (σ). D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo ++ e MI e randômica.

Algoritmo	D(↑)		DB (↓)		S(↑)		R(↑)		I	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
++	0,1029	0,0229	0,9534	0,0013	0,4247	0,0004	0,8667	0,0025	6,1000	1,8947
M1	0,1006	0,0228	0,9532	0,0012	0,4247	0,0004	0,8670	0,0025	7,3000	2,3043
k-Means	0,0937	0,0210	0,9528	0,0011	0,4245	0,0004	0,8677	0,0023	6,4500	2,1324

Com relação aos 4 índices de validação os valores mostrados na Tabela 6.25 estão bastante próximos uns dos outros. Já com relação ao número de iterações, como os

valores apresentados representam a média de 20 execuções, pode ser inferido que com os centroides iniciais fornecidos pelo M1 o k-Means teve, em média, uma iteração a mais para conseguir convergir, quando comparado com o número médio de iterações realizadas com inicializações fornecidas pelo ++ e randômica.

O problema recorrente do M1, o de não conseguir, por meio da abordagem baseada em grid que usa, escolher um conjunto de centroides iniciais, tornou a acontecer com os dados de Seeds, fazendo o algoritmo optar por uma escolha randômica, tornando o seu funcionamento similar ao do k-Means original.

A Tabela 6.26 apresenta os resultados obtidos pelos algoritmos CCIA, SPSS e MS utilizando os dados do domínio de dados Seeds.

Tabela 6.26 (Seeds) Resultados obtidos na execução dos algoritmos CCIA, SPSS e MS pelo Sistema I-kMeans. D: índice Dunn, DB: índice Davies-Bouldin, S: Sillhouette, R: índice Rand, I: Número de iterações realizadas pelo k-Means, até a sua convergência, utilizando as inicializações encontradas pelo CCIA, SPSS e MS.

Algoritmo	D	DB	S	R	I
CCIA	0,08	0,9521	0,4243	0,8693	4
SPSS	0,08	0,9521	0,4243	0,8693	3
MS	0,126	0,9547	0,4252	0,8642	6

Os resultados relacionados aos quatro índices de validação apresentados na Tabela 6.26 são praticamente os mesmos, o que permite inferir que os agrupamentos induzidos pelo k-Means com inicializações fornecidas pelos três algoritmos de inicialização são o mesmo agrupamento. O algoritmo SPSS, entretanto, forneceu o melhor conjunto de centroides iniciais dado que, a partir desses centroides, o k-Means precisou apenas de 3 iterações para convergir. No domínio Seeds, considerando os resultados de ambas tabelas, a 6.25 e a 6.26, fica claro que o algoritmo SPSS foi o que melhor inicializou o k-Means.

6.2.3.3 Considerações sobre os Resultados Obtidos com Dados Artificiais e Dados Reais

Esta seção resume os resultados obtidos com os cinco métodos de inicialização de centroides, nos 12 domínios de dados considerados. Os resultados mostrados na Seção 6.2.3.2 permitem avaliar, dentre os algoritmos de inicialização, aquele que mais contribuiu para a indução, pelo k-Means, de agrupamentos que refletissem a organização original dos dados em grupos, com um menor número de iterações para a sua convergência.

Devido aos conjuntos de dados dos experimentos serem divididos em conjuntos de dados sintéticos e conjuntos de dados reais, os resultados são mostrados separadamente, divididos em duas tabelas.

A Tabela 6.27 apresenta os resultados relativos aos oito conjuntos de dados sintéticos e melhores resultados obtidos nos experimentos.

Tabela 6.27 Melhores resultados dos experimentos com conjunto de dados sintéticos. #k-Means: NI número de iterações do k-Means original, com inicialização randômica, Algoritmo = #R: Algoritmo com melhor desempenho medido via índice Rand, com o valor associado desse índice. e Algoritmo(s) = #Menor NI: Algoritmo e menor número de iterações para k-Means.

Conjunto de Dados	#k-Means NI	Algoritmo = #R	Algoritmo(s) = #Menor NI
Ruspini	2,65	CCIA = 1	CCIA = 0
MS	4,50	SPSS = 0,98	SPSS = 1
Mouse-Like	7,25	CCIA e SPSS = 0,73	CCIA e SPSS = 5
Spherical_6_2	5,15	CCIA = 1	CCIA = 0
LongSquare	12,30	SPSS = 0,94	SPSS = 2
Aggregation	18	CCIA = 0,92	SPSS = 13
3MC	6,95	SPSS = 0,91	CCIA = 0
Teste_M1	1,40	CCIA = 1	CCIA = 0

Analisando os valores mostrados na Tabela 6.27 pode se dizer que os algoritmos CCIA e SPSS, nesta ordem, induziram centroides iniciais mais apropriados, fazendo com que o algoritmo k-Means convergisse com o menor número de iterações. Considerando os oito conjuntos de dados sintéticos utilizados, em cinco deles a inicialização fornecida pelo CCIA promoveu o melhor desempenho do k-Means, relativo à qualidade do agrupamento como, indicam os valores do índice R e fez com que k-Means convergisse com o menor número de iterações.

A Tabela 6.28 apresenta os resultados dos experimentos relativos ao uso dos quatro conjuntos de dados reais e os melhores resultados obtidos nos experimentos.

Tabela 6.28 Melhores resultados dos experimentos com conjunto de dados reais. #k-Means: NI número de iterações do k-Means original, com inicialização randômica, Algoritmo = #R: Algoritmo com melhor desempenho medido via índice Rand, com o valor associado desse índice. e Algoritmo(s) = #Menor NI: Algoritmo e menor número de iterações para k-Means.

Conjunto de Dados	#k-Means NI	Algoritmo = #R	Algoritmo(s) = #Menor NI
Iris	6,25	CCIA = 0,8737	CCIA = 3
Fossil	4,75	MS = 1	MS = 2
Wine	4,7	k-Means = 0,9313	++ = 4,7
Seeds	6,45	SPSS = 0,8693	SPSS = 3

Com exceção dos algoritmos M1 e ++, os demais algoritmos obtiveram bons resultados nos quatros conjuntos de dados mostrados na Tabela 6.28. Os números na tabela evidenciam que os algoritmos CCIA, MS e SPSS forneceram inicializações ao k-Means que promoveram a diminuição do número de iterações realizadas pelo algoritmo, para alcançar convergência, quando comparados com os números alcançados pelo k-Means quando usa o seu procedimento default para inicialização i.e., de escolha randômica.

Como evidenciam os resultados mostrados nas Tabela 6.27 e Tabela 6.28, o CCIA e SPSS são os algoritmos que induziram melhores agrupamentos e diminuíram o número de iteração para k-Means alcançarem as convergências. Considerando os doze conjunto de dados usados nos experimentos conduzidos, o CCIA obtém melhores resultados em seis deles, seguido por SPSS que obteve melhores resultados em cinco deles. Pode se dizer, também, que os dois algoritmos obtiveram melhores resultados nos domínios de dados sintéticos.

6.3 Relato Conciso do Trabalho de Pesquisa Realizado, Considerações e Conclusões

Como evidenciado e comentado em algumas partes desta dissertação, a proposta original do algoritmo de agrupamento k-Means tem uma fase de inicialização dos centroides que, via de regra, é feita de maneira randômica. Várias referências encontradas na literatura apresentam algoritmos propostos com o intuito de fornecer não uma escolha randômica mas sim, uma escolha apropriada que melhore o desempenho final do k-Means, tanto em relação ao agrupamento que induz, quanto ao número de iterações realizadas para a sua indução.

Na fase inicial do desenvolvimento da pesquisa descrita nesta dissertação, durante o levantamento bibliográfico realizado, foi identificado um conjunto de algoritmos de inicialização, todos com o objetivo de colaborar com o resultado final obtido pelo k-Means, por meio de um processo de escolha criteriosa do conjunto de centroides iniciais. Dentre as propostas identificadas foram selecionadas as mais promissoras, tendo como critério, principalmente, o fato de já terem sido citadas em vários trabalhos, apresentarem uma descrição em pseudocódigo do algoritmo proposto e tivessem sido implementadas, com exibição dos resultados obtidos. Dentre os inúmeros algoritmos que se qualificavam, buscando contemplar uma diversidade de técnicas e, também,

considerando o período de tempo disponível para a realização da pesquisa, foram selecionados: (1) k-Means++, devido à relevância, devido aos bons resultados e popularidade desse algoritmo; (2) SPSS, devido ao fato dos autores o considerarem uma melhoria do algoritmo k-Means++; (3) CCIA, devido ao fato de ter uma estratégia híbrida, baseada em densidade e condensação multi-escala; (4) Method-1 por ser baseado em densidade de padrões e abordar o espaço sob a perspectiva de um grid e (5) Maedeh-Suresh, por ser um método baseado em distâncias. Cada um dos cinco algoritmos foi estudado com base em sua descrição, no artigo em que foi proposto, bem como nos comentários/resultados de experimentos contidos em outros trabalhos, que o usaram de uma forma ou outra.

Durante o estudo e entendimento detalhado dos cinco algoritmos, com vistas à investigação da contribuição efetiva de cada um deles, ficou claro que tanto o SPSS quanto o Method-1 sofrem de algumas deficiências. Os problemas relacionados ao SPSS surgem como consequência de sua descrição pouco clara e descuidada, o que promove muitas dúvidas com relação ao processo que o algoritmo tenta descrever. As tentativas de contato com o autor principal não foram bem sucedidas e, via levantamento bibliográfico, não foi possível localizar outras referências que apresentassem uma descrição algorítmica do SPSS; esses fatos todos dificultaram a sua implementação. Já com o algoritmo Method-1, os problemas surgiram pelo fato do algoritmo descrito na referência em que é proposto, deixar de considerar algumas situações limites que poderiam comprometer o resultado obtido pelo algoritmo, como discutido na Seção 5.2. Os problemas relacionados ao Method-1 puderam ser contornados por meio do tratamento das situações limite não consideradas pelo algoritmo, pela implementação do algoritmo, disponibilizada como parte do sistema computacional desenvolvido durante esse trabalho de pesquisa.

O sistema computacional que foi desenvolvido ao longo do trabalho de pesquisa, como apresentado na Seção 6.1, disponibiliza as implementações de cinco algoritmos, e pode ser usado como um ambiente para experimentação com algoritmos de inicialização de centroides em geral *i.e.*, sua arquitetura é flexível à incorporação de implementações de novos algoritmos.

Com relação aos resultados dos experimentos, pode ser inferido que, nos conjuntos de dados selecionados, os algoritmos de inicialização de centroides CCIA e SPSS, foram os que tiveram os melhores resultados, considerando os centroides que calcularam, com vistas à inicialização do k-Means. Os centroides fornecidos por esses

dois algoritmos fez com que o k-Means induzisse bons agrupamentos e diminuísse o número de iterações para convergir, quando comprado com o número de iterações que realiza, quando inicializado via o seu processo randômico default.

Como visto anteriormente, em quatro experimentos com o uso do CCIA, esse algoritmo foi muito bem sucedido no processo de escolha inicial de centroides a serem fornecidos ao k-Means. Foi tão bem sucedido que o k-Means, com os centroides fornecidos pelo CCIA, já satisfazia o critério de convergência, não necessitando de realizar qualquer iteração. Essa situação não se repetiu com qualquer dos outros quatro algoritmos utilizados. É importante mencionar que em AM todos os algoritmos tem um viés e, dependendo do formalismo que empregam e os dados aos quais estão sendo aplicados, o viés que implementam pode favorecer ou não o processo de agrupamento, (*e.g.*, atributos irrelevantes), podem influenciar os centroides iniciais fornecidos pelos métodos investigados, na dependência dos vieses empregados pelos algoritmos, bem como em como os dados estão distribuídos no espaço definido pelos atributos que os caracterizam. O trabalho realizado oferece um panorama detalhado e aprofundado do funcionamento dos cinco algoritmos de inicialização do k-Means e algumas evidências empíricas de desempenho deles em um conjunto reduzido de conjuntos de dados. É importante enfatizar que durante a realização do trabalho tivemos que lidar com situações em que as descrições de alguns dos algoritmos terem sido publicadas de maneira incompleta, com informações ambíguas e dados incorretos. E, em algumas dessas situações, tivemos que recorrer ao autor do algoritmo, em busca de mais informações/esclarecimentos.

6.4 Trabalhos Futuros

Com a finalização do trabalho de pesquisa realizado, algumas possibilidades para futuras investigações ficaram melhor delineadas e, entre elas:

1. Utilizar uma combinação dos centroides encontrados pelos cinco algoritmos pesquisados (*e.g.*, a média aritmética dos valores encontrados pelos algoritmos), e inicializar o k-Means com esses valores.
2. Implementar uma variação do processo descrito em 1, ponderando o desempenho individual dos cinco algoritmos.

3. Refinamento no sistema computacional I-kMeans com o intuito de facilitar o seu uso na realização de experimentos e acompanhamento de resultados, tais como:

3.1 executar k-Means com centroides escolhidos de forma manual;

3.2 rastrear o processo de convergência de k-Means acompanhando cada iteração do algoritmo;

3.3 permitir a escolha do número execuções para algoritmos com inicialização aleatória,

3.4 armazenar a média e desvio padrão de forma automática permitindo a exportação dos resultados para uso posterior.

Referências

[Anderberg 1973] Anderberg, M. R. (1973) Cluster analysis for applications, NY: Academic Press.

[Aggarwal & Reddy 2013] Aggarwal, C. C.; Reddy, C. K. (2013) Data clustering: algorithms and applications, Chapman & Hall/CRC Data Mining and Knowledge Discovery Series, CRC Press.

[Al-Daoud & Roberts, 1996] Al-Daoud, M.; Roberts, S. A. (1996) New methods for the initialization of clusters, *Pattern Recognition Letters*, v. 17, pp. 451-455.

[Altman 1992] Altman, N. S. (1992) An introduction to kernel and nearest-neighbor nonparametric regression, The American Statistician, v. 46, no. 3, pp. 175–185.

[Arlot & Celisse 2010] Arlot, S.; Celisse, A. (2010) A survey of cross-validation procedures for model selection, *Statistics Surveys*, v. 4, pp. 40-79.

[Arthur & Vassilvitskii 2007] Arthur, D.; & Vassilvitskii, S. (2007) K-Means++: the advantages of careful seeding, in Proc. of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms. Society for Industrial and Applied Mathematics, USA, pp. 1027-1035.

[Bentley 1975] Bentley, J. (1975) Multidimensional binary search trees used for associative searching, *Communications of the ACM*, v. 18, no. 9, pp. 509-517.

[Bertini & Nicoletti 2008] Bertini Jr., J. R.; Nicoletti, M. C. (2008) A constructive neural network algorithm based on the geometric concept of barycenter of convex hull, In: Computational Intelligence: Methods and Applications, IEEE Computational Intelligence Society, Poland, pp. 1-12.

[Bishop 2006] Bishop, C. M. (2006) Pattern recognition and machine learning, USA: Springer-Verlag.

[Bandyopadhyay & Maulik 2002] S. Bandyopadhyay and U. Maulik, "Genetic Clustering for Automatic Evolution of Clusters and Application to Image Classification", *Pattern Recognition*, vol.35, pp. 1197-1208, 2002

[Burks *et al.* 2015] Burks, S.; Harrell, G.; Wang, J. (2015) On initial effects of the k-Means clustering, in: Proc. of The 2015 World Congress in Computer Science, Computer Engineering, & Applied Computing, Las Vegas, USA, pp. 200-205.

[Celebi *et al.* 2013] Celebi, M. E.; Kingravi, H. A.; Vela, P. A. (2013) A comparative study of efficient initialization methods for the k-means clustering algorithm, *Expert Systems with Applications*, v. 40, pp. 200-120.

[Celebi 2015] Celebi, M. E. (ed.) (2015) Partitional clustering algorithms, Berlin: Springer-Verlag.

[Chapelle *et al.* 2006] Chapelle, O.; Scholkopf, B.; Zien, A. (2006) Semi-supervised learning, Cambridge, MA:MIT Press.

[Chernoff 1971] Chernoff, H. (1971) The use of faces to represent points in n-dimensional space graphically, Technical Report no. 71, Office of Naval Research, USA.

[Clark & Niblett 1989] Clark, P.; Niblett, T (1989) The CN2 induction algorithm, *Machine Learning*, v. 3, no. 4, pp. 261-283.

[Davies & Bouldin 1979] Davies, D.; Bouldin, D. (1979) A cluster separation measure, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), v. 1, no. 2, pp. 224-227.

[Dua & Karra Taniskidou 2017] Dua, D. and Karra Taniskidou, E. (2017). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[Duda *et al.*, 2001] Duda, R. O.; Hart, P. F.; Stork, D. G. (2001) *Pattern Classification*, USA: John Wiley & Sons, Inc.

[Dunn 1974] Dunn, J.: Well separated clusters and optimal fuzzy partitions, Journal of Cybernetics, v. 4, pp: 95–104.

[Gallant 1990] Gallant, S. (1990) Perceptron-based learning algorithms, *IEEE Transactions on Neural Networks*, v. 1, no. 2, pp. 179-191.

[Gallant 1994] Gallant, S. (1994) Neural network learning and expert systems, London, England: The MIT Press.

[Gionis *et al.* 2007] Gionis, A., H. Mannila, and P. Tsaparas, Clustering aggregation. ACM Transactions on Knowledge Discovery from Data (TKDD), 2007. 1(1): p. 1-30.

[GMUM.r 2018] GMUM.r, Group of Machine Learning Research, Faculty of Mathematics and Computer Science of Jagiellonian Univ., Kraków, Poland [online] http://r.gmum.net/samples/cec.basic.html.

[Gowda & Diday 1992] Gowda, K. C.; Diday, E. (1992) Symbolic clustering using a new similarity measure, IEEE Transactions on Systems, Man, and Cybernetics, v. 22, no. 2, pp. 368-378.

[Halkidi *et al.* 2001] Halkidi, M.; Batistakis, Y.; Vazirgiannis, M. (2001) On clustering validation techniques, Journal of Intelligent Information Systems, v. 17, pp. 107-145.

[Han *et al.* 2012] Han, J.; Kamber, M.; Pei, J. (2012) *Data Mining – Concepts and Techniques*, 3rd. Ed., Amsterdam: Morgan Kaufmann Publishers.

[Handl *et al.* 2004] Handl, Julia, and Joshua Knowles. "Multiobjective clustering with automatic determination of the number of clusters". UMIST, Manchester, Tech. Rep. TR-COMPSYSBIO-2004-02 (2004).

[He *et al.* 2004] He, J.; Jain, M.; Tan, C. L.; Sung, S. Y.; Low, H. B. (2004) Initialization of cluster refinement algorithms: A review and comparative study, in: Proc. of the 2004 IEEE International Joint Conference on Neural Networks, pp. 297-302.

[Jain & Dubes 1988] Jain, A. K., Dubes, R.C. (1988) Algorithms for Clustering Data, Prentice Hall.

[Jain *et al.* 1999] Jain, A. K.; Murty, M. N.; Flynn, P. J. (1999) Data clustering: a review, *ACM Computing Surveys*, v. 31, no. 3, pp. 264-323.

[Jain 2010] Jain, A.K. (2010) Data clustering: 50 years beyond K-Means, *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666.

[Kaufman & Rousseeuw 1990] Kaufman, L.; Rousseeuw, P. (1990) Finding groups in data: an introduction to cluster analysis, Wiley Interscience.

[Kaufman & Rousseeuw 2005] Kaufman, L. and Rousseeuw, P.J. (2005) Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley, Hoboken.

[Khan & Ahmad 2004] Khan, S. S.; Ahmad, A. (2004) Cluster center initialization algorithm for k-Means clustering, *Pattern Recognition Letters*, v. 25, pp. 1293-1302.

[Li 2011] Li, C. S. (2011) Cluster center initialization method for k-Means algorithm over data sets with two clusters, *Procedia Engineering*, v. 24, pp. 324-328.

[Lichman 2013] Lichman, M. (2013) UCI Machine Learning Repository [http://archive.ics.uci.edu/ml], Irvine, CA: University of California, School of Information and Computer Science.

[Liu *et al.* 2010] Liu, Y.; Li, Z.; Xiong, H.; Gao, X.; Wu, J. (2010) Understanding of internal clustering validation measures, 2010 IEEE International Conference on Data Mining, pp. 911-916.

[MacQueen 1967] MacQueen, J. B. (1967) Some methods for classification and analysis of multivariate observations, Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297.

[Maedeh & Suresh 2013] Maedeh, A.; Suresh, K. (2013) Design of efficient k-Means clustering algorithm with improved initial centroids, *International Journal of Engineering and Technology*, v. 5, no. 1, pp. 33-38.

[Maulik & Bandyopadhyay 2002] Maulik, U.; Bandyopadhyay, S. (2002) Performance evaluation of some clustering algorithms and validity indices, *IEEE Transactions on Pattern Analysis and Machine Intelligence* (TPAMI), v. 24, pp. 1650-1654.

[Meila & Heckerman 1998] Meila, M., Heckerman, D., (1998) An experimental comparison of several clustering methods, Microsoft Research Report MSR-TR-98-06, Redmond, WA.

[Mitchell 1997] Mitchell, T. M (1997) Machine Learning, USA: McGraw-Hill.

[Mitra *et al.* 2002] Mitra, P.; Murthy, C. A.; Pal, S. K. (2002) Density-based multiscale data condensation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 24, no. 6, pp. 734-747.

[Nicoletti 1994] Nicoletti, M. C. (1994) Ampliando os limites do aprendizado indutivo de máquina através das abordagens construtiva e relacional, Tese de doutorado, IFSC-USP.

[Nicoletti *et al.* 2010] Nicoletti, M. C.; Bertini Jr., J. R.; Elizondo, D.; Franco, L.; Jerez, J. M. (2010) Constructive neural network algorithms for feedforward architectures suitable for classification tasks, In: Constructive Neural Networks, Studies in Computational Intelligence, D. Elizondo, L. Franco and J. M. Jerez (Eds.), Berlin: Springer-Verlag.

[Pham *et al.* 2004] Pham, D. T.; Dimov, S. S.; Nguyen, C. D. (2004) Selection of k in k-Means clustering, *Mechanical Engineering Science*, v. 219. pp. 103-119.

[Pavan *et al.* 2010] Pavan, K. K.; Rao, A. A.; Rao, A. V. D. and Sridhar, G. R. (2010) Single pass seed selection algorithm for k-Means, *Journal of Computer Science*, v. 6, no. 1, pp. 60-66.

[Pavan *et al.* 2011] Pavan, K. K.; Rao, A. A.; Rao, A. V. D. and Sridhar, G. R. (2011) Robust seed selection algorithm for k-means type algorithms, *International Journal of Computer Science & Information Technology* (IJCSIT), v. 3, no. 5, pp. 147163.

[Peña *et al.* 1999] Peña, J. M.; Lozano, J. A.; Larrañaga, P. (1999) An empirical comparison of four initialization methods for the K-Means algorithm, *Pattern Recognition Letters*, v. 20, pp. 1027-1040.

[Quinlan 1986] Quinlan, J. R. (1986) Induction of decision trees, *Machine Learning*, (1), pp. 81-106.

[Quinlan 1993] Quinlan, J. R. 1993. Programs for Machine Learning. *Morgan Kaufmann Publishers*, Inc. USA: Editorial Office - 2929 Campus Drive, Suite 260, San Mateo, CA 94403.

[Rand 1971] Rand, W. M. (1971) Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical Association, American Statistical Association*, v. 66, no. 336, pp. 846-850.

[Rousseeuw 1987] Rousseeuw, P. (1987) Silhouetters: a graphical aid to the interpretation and validation of cluster analysis, *J. Computational Applied Mathematics*, v. 20, no. 1, pp. 53-65.

[Ruspini 1970] E. H. Ruspini (1970) Numerical methods for fuzzy clustering. Inform. Sci. 2, 319–350.

[Schaffer 1993] Shaffer, C. (1993) Selecting a classification method by cross-validation, *Machine Learning*, v. 13, no. 1, pp. 135-143.

[Su, et al. 2005] M. C. Su, C. H. Chou, and C. C. Hsieh, "Fuzzy C-Means Algorithm with a Point Symmetry Distance," International Journal of Fuzzy Systems, vol. 7, no. 4, pp. 175-181, 2005.

[Theodoridis & Koutroumbas 1999] Theodoridis, S.; Koutroumbas, K. (1999) *Pattern Recognition*, USA: Academic Press.

[Theodoridis & Koutroumbas 2009] Theodoridis , S., & Koutroumbas, K. (2009) *Pattern Recognition*, 4th. Ed.,. USA: Academic Press.

[Witten *at al.* 2011] Witten, I. H.; Frank E.; Hall, M. A. (2011) *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd. Ed., Amsterdam: Morgan Kaufmann Publishers.