Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Ciência da Computação

Análise Comparativa de Métodos de Agrupamentos

Raphael de Oliveira Paiva

JUIZ DE FORA
ABRIL, 2013

Análise Comparativa de Métodos de Agrupamentos

Raphael de Oliveira Paiva

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Ciência da Computação

Orientador: **Custódio Gouvêa Lopes da Motta**

JUIZ DE FORA

ABRIL, 2013

Análise Comparativa de Métodos de Agrupamentos

Raphael de Oliveira Paiva

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Aprovada por:

Custódio Gouvêa Lopes da Motta
DSc em Engenharia Civil COPPE/UFRJ

Raul Fonseca Neto
DSc em Engenharia de Sistemas e Computação COPPE/UFRJ

Carlos Cristiano Hasenclever Borges
DSc em Engenharia Civil COPPE/UFRJ

JUIZ DE FORA

12 DE ABRIL, 2013

Resumo

O presente trabalho apresenta, inicialmente, conceitos relativos à mineração de dados e, em seguida, detalha um estudo comparativo de alguns métodos particionais e baseados em modelo da análise de agrupamento. O objetivo desse estudo é avaliar o comportamento dos métodos quando submetidos a características distintas de bases de dados, variando os números de amostras, de atributos e de classes. Outro interesse está em verificar a influência de técnicas de padronização dos dados nos resultados finais obtidos por cada método de agrupamento. Foram utilizados quatro métodos de agrupamentos: K-means e K-medoid implementados especialmente para este trabalho e K-means e o método de Maximização de Expectativas (EM) disponíveis para uso acadêmico no sistema WEKA. Todos os algoritmos foram submetidos a diversos testes, com seis bases de dados tradicionais, de forma a verificar suas acurácias de predição e seus desempenhos. Finalizando, avalia-se a eficácia dos métodos usados nesse trabalho, bem como, identifica-se quais deles são adequados para cada base de dados em questão, visto que os resultados em análise de agrupamento dependem bastante do problema apresentado.

Palavras-chave: Mineração de dados, análise de agrupamento, K-means, K-medoid, Maximização de Expectativas.

Abstract

This work initially presents concepts related to data mining and then details a comparative study of some partitioning and model-based methods of cluster analysis. The objective of this study is to evaluate the performance of the methods when submitted to different characteristics of databases, varying numbers of samples, attributes and classes. Another interest is to investigate the influence of techniques of data standardization in the final results obtained by each clustering method. We used four clustering methods: K-means and K-medoid implemented especially for this work and K-means and the method of Expectation-Maximization (EM) available for academic use in WEKA system. All algorithms were submitted to various tests with six traditional databases in order to verify their accuracies prediction and their performances. Finally, we evaluate the effectiveness of the methods used in this work, as well as identifies which ones are appropriate for each database in question, since the results in cluster analysis rely heavily on the problem presented.

Keywords: Data-mining, cluster analysis, K-means, K-medoid, Expectation-Maximization.

Agradecimentos

Agradeço primeiramente aos meus queridos pais, Dirce e Sérgio, pela compreensão, paciência e apoio.

Ao Professor Custódio pela orientação, amizade e principalmente pela disponibilidade e atenção oferecida, sem as quais este trabalho não se realizaria.

Aos Professores Raul e Carlos Cristiano, por aceitarem o convite de participar da banca de avaliação deste trabalho.

À todos os meus parentes, pelo apoio e encorajamento.

Aos meus amigos de curso, em especial ao Humberto, Micael, Mui, Nilton, Roberto Nalon e Tárcio pela convivência durante todos esses anos e pelos bons momentos passados juntos.

À todos os Professores do Departamento de Ciência da Computação pelos seus ensinamentos e aos funcionários do curso, que durante esses anos, contribuíram de algum modo para o meu enriquecimento pessoal e profissional.

"O importante é isso: estar pronto para, a qualquer momento, sacrificar o que somos pelo que poderíamos vir a ser."

Charles Du Bois

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

O crescente avanço na tecnologia de geração e armazenamento de dados está produzindo enormes conjuntos de dados em uma grande diversidade de disciplinas científicas em medicina, ciências, engenharias etc. Isso faz com que ferramentas e técnicas tradicionais de análise de dados, muitas vezes não possam ser usadas em conjuntos de dados de tamanho muito grande, ou em razão da natureza não trivial dos dados.

É nesse contexto que entra a mineração de dados, que é o processo de extração automática de conhecimentos em grandes depósitos de dados. É uma tecnologia que combina métodos tradicionais de análise de dados com algoritmos sofisticados. Ela também abre oportunidades interessantes para se explorar e analisar novos tipos de dados.

As técnicas de mineração de dados são organizadas para agir sobre grandes bancos de dados com o intuito de descobrir padrões úteis e desconhecidos. Algumas aplicações são: detecção de fraudes, análise de marketing, retenção de clientes, controle de produção e exploração científica. Essas técnicas também podem fornecer a capacidade de previsão de resultados futuros, como, por exemplo, a previsão de que um cliente recém-chegado a uma loja de departamentos vá gastar mais ou menos do que 100 reais.

Um sistema de mineração de dados tem potencial para gerar milhares e até milhões de padrões ou regras. Porém somente uma pequena fração são de interesse de algum usuário. Para um padrão ser considerado interessante, é preciso:

- Ser facilmente entendido por um ser humano.
- Válido em alguma base de teste com um certo grau de certeza.
- Ser útil.
- Ser original.

Um padrão também é de interesse se ele validar uma hipótese que algum usuário esteja buscando confirmar.

Um padrão interessante representa um conhecimento.

O processo de descoberta de conhecimento em base de dados envolve diversas etapas, destacando-se a seguinte sequência (Fayyad et al, 1996):

1. Consolidação de dados: onde os dados são obtidos a partir de diferentes fontes (arquivos texto, planilhas ou bases de dados) e consolidados numa única fonte.

2. Seleção e pré-processamento: nesta etapa, diversas transformações podem ser aplicadas sobre os dados, como reduzir o número de exemplos, de atributos ou de intervalos de atributos, normalizar valores etc., de forma a obter, no final, um conjunto de dados preparados para utilização dos algoritmos de mineração.

3. Mineração de dados ou DM (*Data Mining*): é a etapa de extração de padrões propriamente dita, onde, primeiramente, é feita a escolha da tarefa de mineração conforme os objetivos desejáveis para a solução procurada, isto é, conforme o tipo de conhecimento que se espera extrair dos dados. A figura 1.1 ilustra as tarefas de mineração destacando-se o ramo que será abordado neste trabalho.

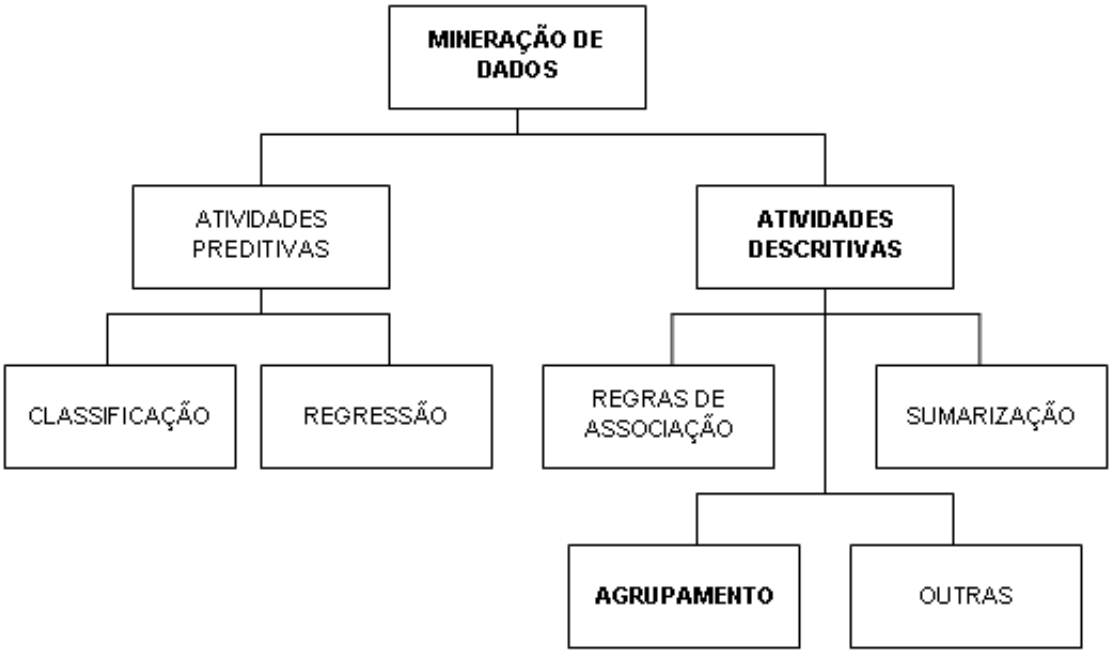


Figura 1.1: Tarefas de Mineração de Dados (Rezende et al, 2003)

As atividades preditivas buscam identificar a classe de uma nova amostra de dados, a partir do conhecimento adquirido de um conjunto de amostras com classes conhecidas. Já as atividades descritivas trabalham com um conjunto de dados que não

possuem uma classe determinada, buscando identificar padrões de comportamento comuns nestes dados.

Em seguida, é escolhido o algoritmo que atenda a tarefa de mineração eleita e que possa representar satisfatoriamente os padrões a serem encontrados. Os algoritmos de mineração mais comuns são: Algoritmos Estatísticos, Algoritmos Genéticos, Árvores de Decisão, Regras de Decisão, Redes Neurais Artificiais, Algoritmos de Agrupamento, Lógica *Fuzzy*.

A mineração de dados é na verdade uma atividade interdisciplinar pela diversidade de tecnologias que podem estar envolvidas. A Figura 1.2 sintetiza os assuntos envolvidos com DM.

Figura 1.2: Assuntos envolvidos com Mineração de Dados (Han e Kamber, 2006)

4. Avaliação e interpretação: nesta etapa são avaliados o desempenho e a qualidade das regras extraídas, bem como verificada a facilidade de interpretação dessas regras.

A utilização do conhecimento obtido no processo de descoberta do conhecimento é realizada através de um sistema inteligente ou de um ser humano como forma de apoio à tomada de decisão.

Entende-se como inteligente um sistema computacional que possui habilidades

inteligentes e sabe como elas modelam tarefas específicas. Entre essas habilidades, está a de usar conhecimento para resolver problemas (Rezende et al, 2003; Motta, 2004).

Alguns desafios motivadores para a mineração de dados são (Tan et al, 2009):

Escalabilidade Para lidar com grandes conjuntos de dados (terabytes ou até petabytes) de forma eficiente, precisa-se de algoritmos escaláveis. Formas comuns de melhorar a escalabilidade incluem o uso de amostragens e o desenvolvimento de algoritmos paralelos e distribuídos, entre outras técnicas.

Alta Dimensionalidade Atualmente é comum encontrar conjuntos de dados com centenas ou milhares de atributos. Técnicas tradicionais desenvolvidas para dados com baixa dimensionalidade muitas vezes não são eficientes para tais dados de alta dimensionalidade. Também é comum a complexidade computacional aumentar rapidamente com o aumento da dimensionalidade (número de características).

Dados Complexos e Heterogênios Os métodos tradicionais de análise de dados muitas vezes lidam com conjuntos de dados que contêm atributos do mesmo tipo, contínuos ou categorizados. Porém nos últimos anos, é cada vez mais comum o aparecimento de objetos de dados mais complexos, necessitando de técnicas que possam lidar com atributos heterogênios. Um exemplo de tipo não tradicional inclui os dados sobre clima que consistem em séries de medidas temporais (temperatura, pressão, etc) em diversas partes da superfície do planeta ao longo dos anos.

Propriedade e Distribuição dos Dados Atualmente é comum encontrar dados na chamada nuvem computacional, ou seja, dados que não estão armazenados em um mesmo local, portanto se encontram distribuídos geograficamente entre fontes pertencentes a múltiplas entidades. Isto requer o desenvolvimento de técnicas distribuídas de mineração de dados, que geram alguns desafios, como: reduzir a quantidade de comunicação necessária para a computação distribuída, consolidar os resultados da mineração a partir de múltiplas fontes e resolver questões de segurança de dados.

1.1 Análise de Agrupamento

A análise de agrupamento trabalha com objetos de dados sem classes definidas. Para se encontrar os padrões, os objetos são agrupados com base no princípio de maximização da similaridade intraclasse e minimização da similaridade interclasse. Isto é, grupos são formados por objetos que possuam alta similaridade entre eles e alta dissimilaridade dos objetos de outros grupos. Cada grupo então pode ser visto como uma classe de objetos, das quais padrões podem ser inferidos.

1.2 Objetivos

O objetivo desse trabalho é a realização de um estudo comparativo entre três métodos de análise de agrupamento: K-means, K-medoid e Maximização de Expectativas (EM). Para o método K-means foi utilizada uma versão disponível no software de mineração de dados WEKA[1], além de uma implementação desenvolvida especificamente para este trabalho. Para o método K-medoid foi utilizada somente uma implementação específica e para o método de Maximização de Expectativas foi utilizada a implementação contida no software WEKA. Todos os métodos foram submetidos a vários testes através de diversas base de dados.

1.3 Organização do Trabalho

Este trabalho está organizado da seguinte forma:

Inicia-se com uma discussão geral da análise de agrupamento de dados (capítulo 2). Descreve-se os diferentes tipos de métodos de agrupamentos e, a seguir, apresenta-se os algoritmos mais usuais.

A seguir, o capítulo 3 trata das implementações desenvolvidas especificamente para este trabalho, destacando a linguagem utilizada, a descrição das estruturas de dados, das bases de dados e outros detalhes.

No capítulo 4 é realizada uma análise comparativa entre os vários métodos tra-

[1]disponível em http://www.cs.waikato.ac.nz/ml/weka/downloading.html

tados. Todos foram testados com seis diferentes base de dados e, em seguida, são apresentados os resultados obtidos.

Por fim, (capítulo 5) temos as considerações finais e as sugestões de trabalhos futuros.

Como complemento dois anexos são incluídos para detalhar:

- Resumo das base de dados por métodos (Anexo A)
- Matrizes de confusão de cada base de dados para cada método (Anexo B)

2 Análise de Agrupamento

O processo de agrupar um conjunto de objetos físicos ou abstratos em classes de objetos similares é chamado agrupamento ou clusterização. O objetivo da análise de agrupamento é buscar entender a estrutura dos dados, através de uma busca por padrões. Para isso, é preciso utilizar a estratégia de maximizar a homogeneidade de objetos dentro de grupos, ao mesmo tempo em que se maximiza a heterogeneidade entre os grupos.

No aprendizado de máquina, agrupamentos é um exemplo de aprendizagem não-supervisionada ou atividade descritiva. Agrupamento e aprendizagem não-supervisionada não se baseiam em classes predefinidas. Por essa razão, agrupamento é uma forma de aprendizado por observação, em vez de aprendizado por exemplos.

A análise de agrupamentos é uma ferramenta útil de análise de dados em muitas situações diferentes. Por exemplo, um pesquisador que tenha coletado dados por meio de um questionário pode se deparar com um grande número de observações que são sem significado a não ser que sejam classificadas em grupos com os quais possa lidar. A análise de agrupamento pode realizar esse procedimento de redução de dados objetivamente pela redução da informação de uma população inteira ou de uma amostra para a informação sobre subgrupos específicos e menores. Por exemplo, nos negócios o agrupamento pode ajudar profissionais do marketing a descobrirem grupos distintos de clientes em suas bases de dados e caracterizar esse grupos com base nos padrões de compra. Na biologia, pode ser usada para derivar taxonomias de plantas e animais categorizando genes de funções similares. Na tecnologia da informação, pode ser usada para classificar documentos da web com o objetivo de descobrir informações. Outras aplicações são: reconhecimento de padrões, análise de dados e processamento de imagens. (Hair et al, 2005; Han e Kamber, 2006)

Agrupamento também é chamado de segmetação de dados em algumas aplicações por particionar grandes bancos de dados em grupos de acordo com sua similaridade. Também pode ser usado para a detecção de *outliers* (valores que estão "distantes" de qualquer grupo), quando os mesmos forem mais interessantes que os grupos. Por exemplo,

na detecção de invasões, onde um comportamento anormal é mais interessante que os padrões e também detectando fraudes em cartão de créditos, onde casos de gastos caros e frequentes podem indicar uma possível atividade fraudulenta.

O agrupamento é um campo de pesquisa bastante desafiador no qual suas aplicações demandam requisitos especiais. Abaixo temos requisitos típicos de agrupamento na mineração de dados (Han e Kamber, 2006):

- **Escalabilidade**: Muitos algoritmos de agrupamento funcionam muito bem em pequenos conjuntos de dados contendo pouco mais de centenas de objetos. Entretanto, grandes banco de dados, podem conter até milhões de objetos. Agrupar uma amostra de um grande banco de dados pode gerar resultados tendenciosos. Algoritmos altamente escaláveis são necessários.

- **Habilidade de lidar com diferentes tipos de atributos**: Vários algoritmos são projetados para agrupar dados númericos. Entretanto, algumas aplicações podem requisitar o agrupamento de outros tipos de dados, como binários, nominais e dados ordinais, ou uma mistura desse tipos.

- **Descoberta de grupos com formatos arbitrários**: Muitos algoritmos de agrupamentos definem grupos baseados nas medidas de distância Euclidiana ou Manhattan. Algoritmos baseados nessas medidas de distância tendem a achar grupos esféricos com tamanho e densidade similares. Todavia, um grupo pode possuir qualquer formato. É importante desenvolver algoritmos que podem detectar grupos de tamanhos arbitrários.

- **Requisitos mínimos de conhecimento do assunto para determinar parâmetros de entrada**: Vários algoritmos exigem do usuário a entrada de certos parâmetros na análise de agrupamento (por exempo, o número de grupos desejado). Os resultados do agrupamento podem ser bastante sensíveis aos parâmetros escolhidos. Parâmetros são geralmente difíceis de se determinar, especialmente em base de dados contendo objetos com muitas dimensões. Essa situação faz com que a qualidade do agrupamento seja difícil de controlar.

- **Habilidade para lidar com dados com ruídos**: A maioria dos banco de dados do mundo real contém *outliers*, valores faltando, desconhecidos ou errados. Alguns algoritmos são sensíveis a esses dados e podem conduzir a grupos de baixa qualidade.

- **Agrupamento incremental e insensibilidade à ordem dos registros de entrada**: Alguns algoritmos de agrupamento não podem incorporar novos dados (por exemplo, atualização de banco de dados) à grupos já criados e por conta disso, devem determinar um novo agrupamento a partir do zero. Também podem ser sensíveis à ordem de entrada dos dados. Isto é, com um mesmo conjunto de dados, um algoritmo provalvente retornará grupos completamente diferentes dependendo da ordem de entrada dos objetos. É importante desenvolver algoritmos incrementais e que são insensíveis à ordem de entrada.

- **Alta dimensionalidade**: Um banco de dados ou um *data warehouse* podem conter diversas dimensões (atributos). Vários algoritmos são bons ao lidar com dados de baixa dimensionalidade, envolvendo somente duas ou três dimensões. Os olhos humanos são bons em julgar a qualidade de um agrupamento para até três dimensões. Achar grupos em um espaço de alta dimensão é desafiador, especialmente considerando que tais dados podem ser esparsos e altamente assimétricos.

- **Agrupamento baseado em restrições**: Aplicações do mundo real podem necessitar de uma análise de agrupamentos sob vários tipos de restrições. Suponha que seu trabalho seja escolher locais para um dado número de caixas eletrônicos em uma cidade. Para decidir sobre isso, você pode agrupar domicílios considerando as restrições, tais como rios da cidade e redes rodoviárias, e do tipo e número de clientes por grupo. Uma tarefa desafiadora é achar grupos de dados com bom comportamento que satisfazem certas restrições.

- **Interpretabilidade e usabilidade**: Usuários esperam que os resultados do agrupamento sejam interpretáveis, compreensíveis e utilizáveis. Isto é, o agrupamento precisa estar de acordo com interpretações semânticas específicas e suas aplicações. É importante estudar como o objetivo de uma aplicação pode influenciar na seleção das características e métodos do grupo.

2.1 Tipos de Dados em Análise de Grupos

Os algoritmos de agrupamento principais normalmente operam sobre uma das duas estruturas de dados abaixo:

- Matriz de dados: Representa n objetos, por exemplo, pessoas, com p variáveis (atributos), tais como idade, altura, peso, sexo etc. A estrutura é da forma de uma tabela relacional, ou matriz n x p (n objetos x p variáveis):

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix} \tag{2.1}$$

- Matriz de distâncias: Armazena um conjunto de distâncias que estão disponíveis para todos os pares de n objetos. Geralmente é representada por uma tabela n x n:

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \cdots & \cdots & 0 \end{bmatrix} \tag{2.2}$$

  onde $d(i,j)$ é a medida da diferença ou distância entre os objetos i e j. Em geral $d(i,j)$ é um número não negativo que é próximo de zero quando os objetos i e j são bastante similares ou "pertos" entre si, e vai aumentando conforme mais diferentes os objetos se tornam. Como $d(i,j) = d(j,i)$, e $d(i,i) = 0$, temos a matriz apresentada em (2.2).

Vários algoritmos de agrupamento usam uma matriz de distâncias. Porém, caso os dados estejam representados como uma matriz de dados, é possível transformá-los numa matriz de distâncias antes de aplicar estes algoritmos.

Os tipos de variáveis mais comuns usadas para caracterizar os objetos a serem agrupados são:

2.1.1 Variáveis Intervalares

São medidas contínuas sobre uma escala linear. Exemplos comuns são: peso e altura, latitude e longitude.

Segundo Han e Kamber (2006) a unidade de medida usada pode afetar a análise de agrupamento. Por exemplo, alterar uma medida de metros para centímetros de uma altura, pode gerar agrupamentos bastante diferentes. Em geral, expressar uma variável em unidades menores irá gerar um maior intervalo de valores para a variável, e, assim, um efeito maior no resultado do agrupamento. Para ajudar a evitar dependências pelas unidades de medidas, os dados podem ser padronizados. Medidas de padronização tem o objetivo de fazer com que todas as variáveis tenham um mesmo peso na análise. Isto é particularmente útil quando não temos conhecimento prévio sobre os dados. Entretanto em alguns casos, os usuários podem querer intencionalmente oferecer mais peso para certos conjuntos de variáveis que outras. Por exemplo, ao agrupar candidatos de um time de basquete, pode-se preferir valorizar mais a variável altura.

Padronizando Dados de uma Variável

Para padronizar medidas, uma escolha é converter a medida original para uma váriavel sem escala. Existem vários métodos para padronizar uma variável, abaixo destacamos dois deles:

Escore-z: O método escore-z é executado através dos passos abaixo (Han e Kamber, 2006):

1. Calcular o desvio absoluto médio, s_f:

$$s_f = \frac{1}{n}(|x_{1f} - m_f| + |x_{2f} - m_f| + \cdots + |x_{nf} - m_f|), \tag{2.3}$$

onde $x_{1f}, \ldots, x_{nf}$ são n medidas da variável f, e m_f é a média aritmética de f, isto

é $m_f = \frac{1}{n}(x_{1f} + x_{2f} + \cdots + x_{nf})$.

2. Calcular o escore-z:

$$z_{if} = \frac{x_{if} - m_f}{s_f}, \quad i = \overline{1, \ldots, n} \tag{2.4}$$

O desvio absoluto médio, s_f, é mais robusto a *outliers* que o desvio padrão, σ_f. Ao computar s_f, os desvios da média (ex.$|x_{if} - m_f|$) não são elevados ao quadrado, por isso o efeito dos *outliers* ficam menos evidentes.

Normalização linear: A normalização linear transforma os dados para uma medida sem escala que se encontrará num pequeno intervalo, por exemplo: entre 0 e 1. Para calcular a normalização linear, v, de uma variável, temos que usar a seguinte fórmula:

$$v_{if} = \frac{x_{if} - min}{max - min}, \tag{2.5}$$

onde min e max correspondem, respectivamente, ao menor e maior valor encontrado para a variável em questão.

A padronização pode ser útil ou não em uma aplicação particular. Assim a escolha de quando usar deve ser deixada para o usuário.

Após o uso ou não da padronização, a dissimilaridade (ou distância) entre os objetos descritos por variáveis intervalares é basicamente computada através da distância entre pares de objetos. A medida mais comum de distância é a distância Euclidiana, que é definida como:

$$d(i, j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}, \tag{2.6}$$

onde $i = (x_{i1}, x_{i2}, \cdots, x_{in})$ e $j = (x_{j1}, x_{j2}, \cdots, x_{jn})$ são dois objetos n-dimensionais.

Outra medida conhecida é a distancia Manhattan, definida como:

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}| \tag{2.7}$$

Ambas satisfazem os seguintes requisitos matemáticos de uma função de distância:

1. $d(i, j) \geq 0$: Distância é um número não negativo.

2. $d(i, i) = 0$: A distância de um objeto para ele mesmo é zero.

3. $d(i, j) = d(j, i)$: Distância é uma função simétrica.

4. $d(i, j) \leq d(i, h) + d(h, j)$: Ir diretamente do objeto i para o objeto j no espaço é menor do que atingir j indiretamente por meio de um objeto h (inequação triangular).

2.1.2 Variáveis Binárias

Uma variável binária tem somente dois estados: 0 ou 1, onde 0 significa que a variável está ausente, e 1 significa sua presença. Por exemplo, dada a variável "sedentarismo" descrevendo um paciente, 1 indica que o paciente é sedentário, enquanto 0 indica que ele não é. Tratar variáveis binárias como variáveis intervalares pode levar a resultados equivocados. Assim métodos específicos para dados binários são necessários para computar a dissimilaridade.

Segundo Han e Kamber (2006), uma abordagem para calcular a dissimilaridade envolve o uso de uma matriz de distâncias. Se todas as variáveis binárias tiverem o mesmo peso, teremos uma tabela de contingência 2 x 2, como a tabela 2.1, onde q é o número de variáveis iguais a 1 para ambos os objetos i e j, r é o número de variáveis iguais a 1 para o objeto i mas iguais a 0 para j, s é o número de variáveis iguais a 0 para i mas iguais a 1 para j, e t é o número de variáveis iguais a 0 para ambos os objetos. O número total de variáveis é p, onde $p = q + r + s + t$.

Tabela 2.1: Tabela de contingência para variáveis binárias (Han e Kamber, 2006)

		Objeto j		
		1	0	soma
Objeto i	1	q	r	$q + r$
	0	s	t	$s + t$
	soma	$q + s$	$r + t$	p

A dissimilaridade entre os objetos i e j, é então definida pela equação abaixo:

$$d(i, j) = \frac{r + s}{q + r + s + t} \tag{2.8}$$

2.1.3 Variáveis Nominais

São uma generalização das variáveis binárias pois podem possuir mais de dois estados. Por exemplo "estado civil" é uma variável nominal que pode conter cinco estados: solteiro, casado, separado, divorciado e viúvo.

Os estados podem ser representados por letras, símbolos ou um conjunto de inteiros. Note que os inteiros são usados somente para identificação dos dados, não indicando nenhuma ordem específica.

A dissimilaridade entre dois objetos i e j pode ser computado com base na proporção de desemparelhamentos:

$$d(i,j) = \frac{p-m}{p}, \tag{2.9}$$

onde m é o número de correspondências (número de variáveis nas quais i e j são do mesmo estado), e p o número total de variáveis.

2.1.4 Variáveis Ordinais

Uma variável ordinal discreta lembra a variável nominal, exceto pelo fato que os estados da variável ordinal estão ordenados em uma sequência com significado. A variável de tipo ordinal resulta da operação de ordenar por postos. Assim, estabelece-se uma ordem hierárquica entre as categorias. A ordem resulta da distinção dos elementos de acordo com o maior ou menor grau com que possuem determinada característica. Por exemplo, a variável "nível socioeconômico" pode ser hierarquizada conforme abaixo:

- nível alto;
- nível médio;
- nível baixo.

Nesse caso, tem-se uma variável ordinal que implica uma ordem quantitativa, numérica, só em termos de maior ou menor, sem se estabelecer com precisão quanto mais alto ou mais baixo é o nível socioeconômico de uma ou outra categoria.

Variáveis ordinais também podem ser obtidas da discretização de quantidades intervalares, dividindo a amplitude dos valores em um número finito de classes.

Calcular a dissimilaridade em variáveis ordinais é parecido com o processo das variáveis intervalares. Suponhamos que f seja uma variável de um conjunto de variáveis ordinais que descrevem n objetos. O cálculo da dissimilaridade envolve os seguintes passos:

1. O valor de f para o i-ésimo objeto é x_{if}, e f tem M_f estados ordenados, representando as posições $1, \ldots, M_f$. Substitui-se cada x_{if} por sua posição correspondente, $r_{if} \in \{1, \ldots, M_f\}$.

2. Como cada variável ordinal pode ter quantidades diferentes de estados, é geralmente necessário mapear a amplitude de cada variável para o intervalo [0,1], assim cada variável terá um peso igual. Para isso deve-se substituir a posição r_{if} do i-ésimo objeto na f-ésima variável por:

$$z_{if} = \frac{r_{if} - 1}{M_f - 1} \tag{2.10}$$

3. A dissimilaridade pode então ser computada usando qualquer medida de distância descrita na subseção 2.1.1 sobre variáveis intervalares, usando z_{if} para representar o valor f para o i-ésimo objeto.

2.1.5 Variáveis de Razão

Fazem medição positiva numa escala não-linear, como a escala exponencial, seguindo aproximadamente a seguinte fórmula:

$$Ae^{Bt} \quad \text{ou} \quad Ae^{-Bt}, \tag{2.11}$$

onde A e B são constantes positivas, e t costuma representar o tempo. Alguns exemplos incluem o crescimento de uma cultura bacteriana e o decaimento de um elemento radioativo.

Existem três métodos para o cálculo da distância de dissimilaridade entre objetos deste tipo (Han e Kamber, 2006):

1. Tratar variáveis de razão como variáveis intervalares. Entretanto, esta não é geralmente uma boa escolha porque provavelmente haverá uma distorção da escala.

2. Aplicar uma transformação logarítmica em uma variável de razão f que possui valor x_{if} para o objeto i usando a fórmula $y_{if} = \log(x_{if})$. O valor y_{if} então pode ser tratado como uma variável intervalar. Dependendo da definição da variável e da aplicação outras transformações podem ser aplicadas.

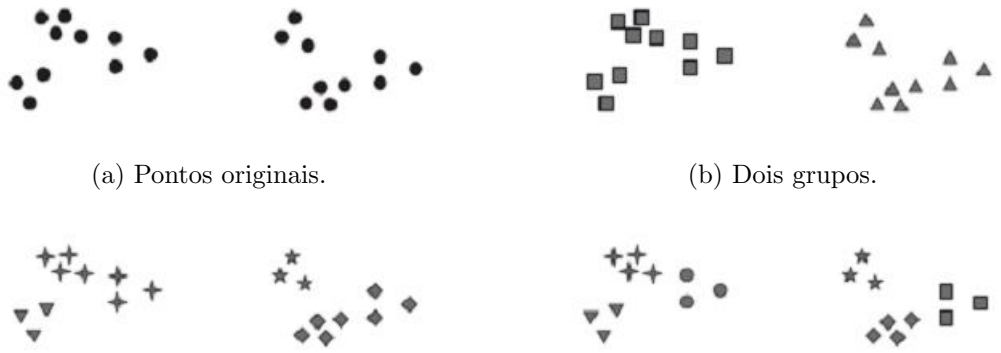
3. Tratar x_{if} como dado ordinal e tratar seus ranks como valores intervalares.

Os dois últimos métodos são mais efetivos, embora a escolha do método a ser utilizado dependa da aplicação.

2.2 Tipos de Grupos

Nesta seção será distinguido os diferentes tipos de grupos na análise de agrupamento. Segundo Tan et al (2009) os grupos se diferem da seguinte forma:

- **Hierárquico versus Particional**: A distinção mais comumente discutida entre diferentes tipos de agrupamentos é verificar se o conjunto de grupos está aninhado ou não. Ou ainda, em terminologia mais tradicional, se é hierárquico ou particional. Um agrupamento particional é simplesmente uma divisão do conjunto de objetos de dados em subconjuntos (grupos) não interseccionados de modo que cada objeto de dado esteja exatamente em um subconjunto. Individualmente, cada conjunto de grupos na figura 2.1 é um agrupamento particional.



(a) Pontos originais. (b) Dois grupos.

(c) Quatro grupos. (d) Seis grupos.

Figura 2.1: Formas diferentes de agrupamento do mesmo conjunto de pontos. (Tan et al, 2009)

Se for permitido que os grupos tenham subgrupos, então tem-se um agrupamento hierárquico, que é um conjunto de grupos aninhados organizados como uma árvore. Cada nodo (grupo) da árvore (exceto pelos nodos folha) é a união dos seus filhos (subgrupos) e o raiz da árvore é o grupo contendo todos os objetos. Muitas vezes, mas nem sempre, as folhas da árvore são grupos únicos de objetos de dados individuais. Permitindo que grupos sejam aninhado, então uma interpretação da figura 2.1a é que ela possui dois subgrupos (figura 2.1b), cada um dos quais, por sua vez, possui três subgrupos (figura 2.1d). Os grupos mostrados na figura 2.1, quando considerados nessa ordem, também formam um agrupamento hierárquico (aninhado) com, respectivamente, 1, 2, 4 e 6 grupos em cada nível. Finalmente, deve-se observar que um agrupamento hierárquico pode ser visto como uma sequência de agrupamentos particionais e um agrupamento particional pode ser obtido pegando-se qualquer membro dessa sequência. Por exemplo, cortando a árvore hierárquica em um determinado nível.

- **Exclusivo versus Interseccionado versus Difuso**: Os agrupamentos mostrado na figura 2.1 são todos exclusivos, já que atribuem cada objeto a um único grupo. Há muitas situações nas quais um ponto poderia ser colocado em mais de um grupo, e estas situações são melhor abordadas pelo agrupamento não exclusivo. No sentido mais geral, um agrupamento não exclusivo ou interseccionado é usado para refletir o fato de que um objeto pode pertencer simultaneamente a mais de um grupo (classe). Por exemplo, uma pessoa em uma universidade pode tanto ser um aluno matriculado quanto um funcionário da universidade. Um agrupamento não exclusivo também é muitas vezes usado quando, por exemplo, um objeto está "entre" dois ou mais grupos e pode ser atribuído a qualquer um desses grupos. Imagine um ponto entre dois dos grupos da figura 2.1. Em vez de fazer uma atribuição um pouco arbitrária do objeto a um único grupo, ele é colocado em todos os grupos "igualmente bons".

  No agrupamento difuso, considera-se que cada objeto pertence a cada grupo, através de um peso que pode variar entre 0 (não pertence totalmente) e 1 (pertence totalmente). Em outras palavras, os grupos são tratados como conjuntos difusos. (Matematicamente, um conjunto difuso é aquele no qual um objeto pertence a qualquer

conjunto com um peso que varia entre 0 e 1. No agrupamento difuso, muitas vezes impõe-se a restrição adicional de que a soma dos pesos de cada objeto deve ser igual a 1.) De forma semelhante, técnicas de agrupamento probabilísticas calculam a probabilidade com a qual cada ponto pertence a cada grupo e estas probabilidades também devem somar 1. Devido às probabilidades ou pesos de ser membro de um objeto somarem 1, um agrupamento probabilístico ou difuso não aborda verdadeiramente situações multiclasses, como no caso de um funcionário aluno, onde um objeto pertence a múltiplas classes. Em vez disso, estas abordagens são as mais apropriadas para evitar a arbitrariedade de atribuir um objeto a apenas um grupo quando pode estar próximo de vários. Na prática, um agrupamento muitas vezes é convertido em um agrupamento exclusivo atribuindo-se cada objeto ao grupo no qual sua probabilidade ou seu peso de ser membro for mais alto.

- **Completa versus Parcial**: Um agrupamento completo atribui cada objeto a um grupo, enquanto que um agrupamento parcial não. A motivação para um agrupamento parcial é que alguns objetos no conjunto de dados não pertencem a grupos bem definidos. Muitas vezes objetos no conjunto de dados podem representar ruídos, *outliers* ou "informações desinteressantes". Por exemplo, algumas matérias de jornais podem compartilhar uma técnica comum, como o aquecimento global, enquanto que outras são mais genéricas ou únicas. Assim, para descobrir os tópicos importantes que sejam altamente relacionados, pode-se querer pesquisar apenas grupos de documentos que estejam altamente relacionados por um tema comum. Em outros casos, um agrupamento completo dos objetos é desejado. Por exemplo, uma aplicação que use agrupamentos para organizar documentos para navegação precisa garantir que todos os documentos possam ser navegados.

2.3 Métodos Particionais

Segundo Han e Kamber (2006), dada uma base de dados de n objetos ou tuplas, um método particional constrói k partições dos dados, onde cada partição representa um grupo e $k \leq n$. Isto é, os dados são classificados em k grupos, os quais juntos satisfazem

as seguintes condições:

- cada grupo deve conter pelo menos um objeto.
- cada objeto só pode pertencer a um grupo.

Dado o número de partições a criar, k, um método particional cria uma partição inicial. Então utiliza de uma técnica de relocação iterativa que tenta melhorar o particionamento através da troca do objeto de um grupo para outro. O critério geral de um bom particionamento é que objetos em um mesmo grupo estão "pertos" ou relacionados entre si, enquanto que objetos de um grupo diferente estão "distantes" ou são bastante diferentes. Existem vários tipos de critérios para julgar a qualidade de um agrupamento.

Para atingir a otimização global num agrupamento baseado em partições seria preciso enumerar exaustivamente todas as possibilidades de partições, o que muitas vezes é inviável computacionalmente. Para contornar esse problema, a maioria das aplicações adota alguns poucos métodos heurísticos populares, como o algoritmo K-means, onde cada grupo é representado pelo valor médio dos objetos no grupo e o algoritmo K-medoid, onde cada grupo é representado por um de seus objetos localizados próximo ao centro do grupo. Esses métodos heurísticos de agrupamento funcionam bem para encontrar grupos globulares em pequenas e médias base de dados. Para encontrar grupos de forma complexas e agrupar grandes quantidades de dados, os métodos particionais precisam ser extendidos.

2.3.1 O Método K-Means

A técnica de agrupamento K-means é simples. Primeiro são escolhidos k centroides iniciais, onde k é um parâmetro especificado pelo usuário, a saber, o número de grupos desejado. A seguir, cada ponto é atribuído ao centroide mais próximo e cada coleção de pontos atribuídos a um centroide é um grupo. O centroide de cada grupo é então atualizado, baseado na média dos pontos do grupo. Os passos de atribuição e atualização são repetidos até que nenhum ponto mude de grupo ou, equivalentemente, até que os centroides permaneçam os mesmos. (Tan et al, 2009)

O K-means é formalmente descrito pelo Algoritmo 1. A operação do K-means é

Algoritmo 1: Algoritmo K-means

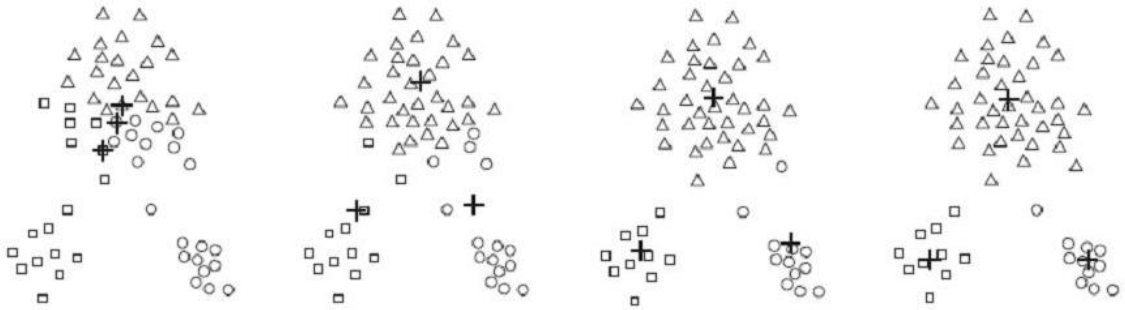
Entrada:

- k: Número de grupos
- D: Base de dados contendo n objetos

Saída: Um conjunto de k grupos

1. Selecione k pontos como centroides iniciais.
2. **repita**
3. Forme k grupos atribuindo cada ponto ao seu centroide mais próximo.
4. Recalcule o centroide de cada grupo.
5. **até que** os centroides não mudem.

ilustrada na figura 2.2, que mostra como, começando de três centroides, os grupos finais são encontrados em quatro passos de atualização. Nestas e em outras figuras exibindo agrupamento K-means, cada subfigura mostra os centroides iniciais da iteração e a atribuição dos pontos a esses centroides. Os centroides são indicados pelo símbolo "+". Todos os pontos que pertencem ao mesmo grupo têm a mesma forma de marcador.



(a) Iteração 1 (b) Iteração 2 (c) Iteração 3 (d) Iteração 4

Figura 2.2: Usando o algoritmo K-means para encontrar três grupos nos dados de exemplo. (Tan et al, 2009)

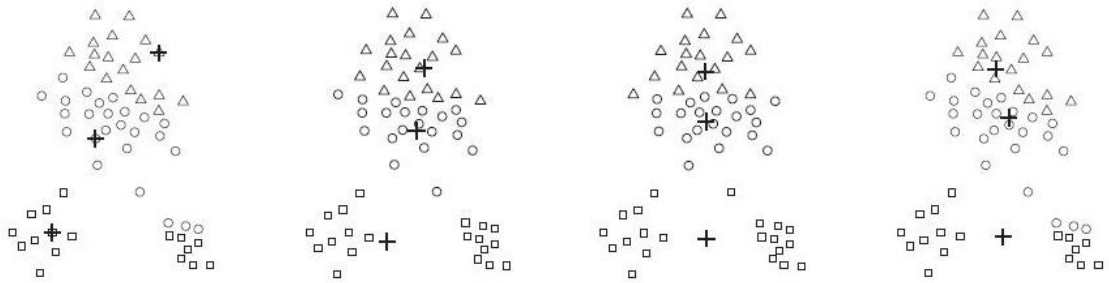
Atribuindo Pontos ao Centroide Mais Próximo

Para atribuir um ponto ao centroide mais próximo, é necessário uma medida de proximidade que quantifique a noção de "mais próximo" para os dados específicos em consideração. A distância Euclidiana é usada frequentemente para pontos de dados no espaço

Euclidiano. Entretanto, essa escolha vai depender do tipo de dados utilizado. Uma preocupação, é que as medidas de semelhanças sejam relativamente simples, pois o algoritmo calcula repetidamente a semelhança de cada ponto com cada centroide. Contudo, existem variações do K-means que são capazes de lidar com cálculos mais complexo para a semelhança mantendo o algoritmo viável em termos de complexidade de tempo.

Escolhendo Centroides Iniciais

Escolher os centroides iniciais apropriados é a etapa chave do procedimento K-means básico. Uma abordagem comum é escolher os centroides iniciais aleatoriamente, mas os grupos resultantes são frequentemente pobres. Na figura 2.3 temos o exemplo de como um centroide inicial pobre pode gerar resultados indesejados.



(a) Iteração 1 (b) Iteração 2 (c) Iteração 3 (d) Iteração 4

Figura 2.3: centroides iniciais pobres para K-means. (Tan et al, 2009)

Segundo Tan et al (2009), uma técnica que pode superar o problema de escolher centroides iniciais, é pegar uma amostra de pontos e agrupá-los usando uma técnica de agrupamento hierárquico, o qual gera grupos aninhados. k grupos são extraídos do agrupamento hierárquico e os centroides desses grupos são usados como centroides iniciais. Esta abordagem muitas vezes funciona bem mas é prática somente se a amostra for relativamente pequena e k for relativamente pequeno comparado com o tamanho da amostra.

Outra abordagem para selecionar centroides iniciais, é selecionar o primeiro ponto aleatoriamente ou escolher o centroide de todos os pontos da base de dados. A seguir, para cada centroide inicial sucessivo, selecionar o ponto que estiver mais distante de qualquer um dos centroides iniciais já selecionados. Desta forma, obtemos um conjunto

de centroides iniciais que certamente sejam não apenas selecionados aleatoriamente mas também bem separados. Infelizmente, tal abordagem pode selecionar *outliers*, em vez de pontos em regiões densas (grupos). Além disso, é custoso calcular o ponto mais distante do conjunto corrente de centroides iniciais.

Complexidade de Espaço e de Tempo

Os requisitos de espaço para o K-means são modestos porque apenas os pontos de dados e centroides são armazenados. Especificamente, o armazenamento requerido é $O((m+k)*n)$, onde m é o número de pontos e n é o número de atributos. Os requisitos de tempo para K-means também são moderados, basicamente linear no número de pontos de dados. Em especial, o tempo necessário é $O(I*k*m*n)$, onde I é o número de iterações necessárias para a convergências. Como I é frequentemente pequeno e pode geralmente ser limitado com segurança, já que a maioria das mudanças geralmente ocorre nas primeiras iterações. Portanto o K-means é linear em m, o número de pontos, e é eficiente assim como simples desde que k, o número de grupos, seja significativamente menor que m.

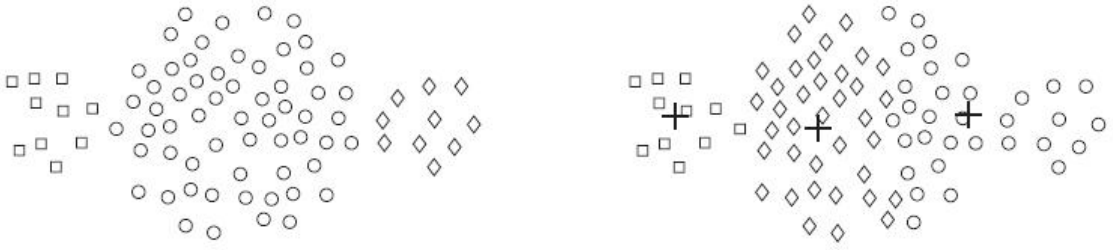
Elementos Externos

Elementos externos, ou *Outliers*, podem gerar distorção nos grupos, alterando significativamento o centroide do grupo ao qual os mesmos foram atribuídos. Por causa disso, muitas vezes é útil descobrí-los e eliminá-los antecipadamente. É importante, entretanto, perceber que há determinadas aplicações de agrupamento para as quais os elementos externos não devem ser eliminados. Por exemplo, em análise financeira, elementos externos aparentes, como clientes que raramente são lucrativos, podem ser os pontos mais interessantes.

Diferentes Tipos de Grupos

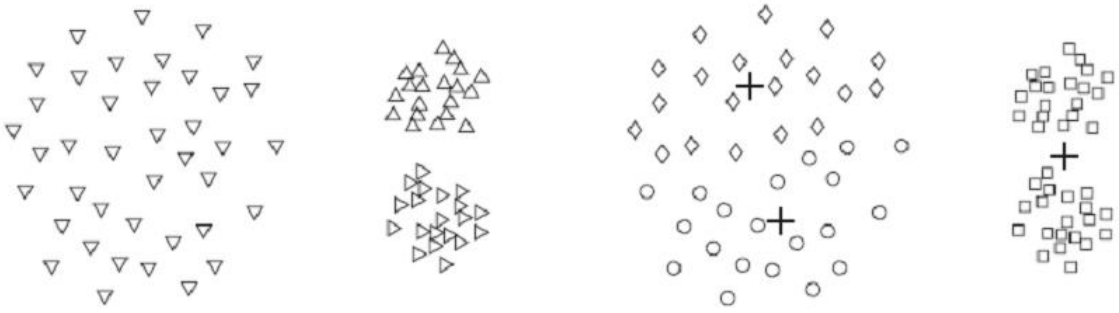
O K-means e suas variações têm um número de limitações com respeito a encontrar diferentes tipos de grupos. Em especial, K-means possui dificuldades para detectar os grupos "naturais", quando os grupos têm formas não esféricas ou tamanhos ou densidades muito diferentes. Isto é ilustrado pelas figuras 2.4, 2.5 e 2.6. Na figura 2.4, o K-means não

consegue encontrar os três grupos naturais porque um dos grupos é muito maior do que os outros dois e, assim, o grupo maior é dividido, enquanto que um dos grupos menores é combinado com uma parte do grupo maior. Na figura 2.5, o K-means não encontra os três grupos naturais porque os dois grupos menores são muito mais densos do que o grupo maior. Finalmente, na figura 2.6, o K-means descobre dois grupos que misturam partes dos dois grupos naturais porque o formato dos grupos naturais não é globular. (Tan et al, 2009)



(a) Pontos originais. (b) Três grupos de K-means.

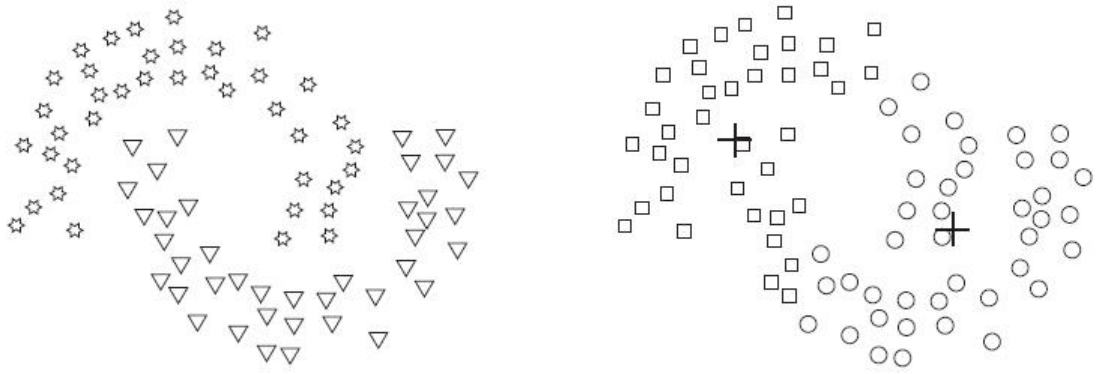
Figura 2.4: K-means com grupos de tamanhos diferentes. (Tan et al, 2009)



(a) Pontos originais. (b) Três grupos de K-means.

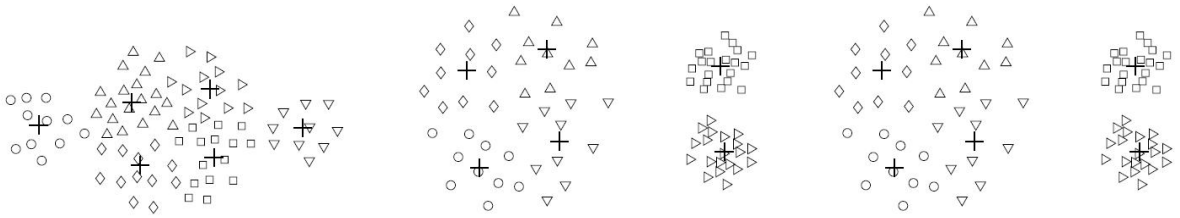
Figura 2.5: K-means com grupos de densidade diferentes. (Tan et al, 2009)

É possível superar esse problemas se o usuário estiver disposto a aceitar um agrupamento que divida os grupos naturais em um certo número de subgrupos. A figura 2.7 mostra o que acontece aos três conjuntos de dados anteriores se forem considerados seis grupos em um conjunto onde exista apenas dois ou três grupos. Cada grupo menor é puro no sentido de que contém apenas pontos de um dos grupos naturais.



(a) Pontos originais. (b) Dois grupos de K-means.

Figura 2.6: K-means com grupos não globulares. (Tan et al, 2009)



(a) Tamanhos diferentes. (b) Densidade diferentes. (c) Forma não esférica.

Figura 2.7: Usando K-means para encontrar grupos que sejam subgrupos dos grupos naturais. (Tan et al, 2009)

Pontos Fortes e Fracos

O K-means é simples e pode ser usado para um ampla variedade de tipos de dados. Também é bastante eficiente, embora múltiplas execuções sejam realizadas com frequência. O K-means não é apropriado para todos os tipos de dados. Ele não pode lidar com grupos não globulares ou de tamanhos e densidades diferentes, embora geralmente consiga encontrar subgrupos puros se um número grande o suficiente de grupos for especificado. (Tan et al, 2009)

A detecção e remoção de elementos externos pode auxiliar significativamente em tais situações.

Finalmente, o K-means é restrito a dados para os quais exista uma noção de um centro (centroide). Uma técnica relacionada, o agrupamento K-medoid, não tem esta restrição, porém é mais custoso.

2.3.2 O Método K-Medoid

Ao contrário do K-means, que utiliza o valor médio dos objetos de um grupo como ponto de referência, K-medoid utiliza objetos verdadeiros para representar os grupos, usando um objeto representante por grupo. Cada objeto restante, é então agrupado com o objeto representante mais similar. O método é então executado baseando-se no princípio de minimização da soma das distâncias entre cada objeto e seu ponto de referência correspondente. No geral, o algoritmo itera até que, enventualmente, cada objeto representante seja o medoide, ou seja, o objeto mais central de seu grupo. (Han e Kamber, 2006)

Os objetos representantes iniciais são escolhidos aleatoriamente. O processo iterativo de troca de um objeto representante por um objeto não-representante continua enquanto a qualidade do agrupamento esteja melhorando.

Funcionamento do Algoritmo

Sejam $M_1, M_2, \ldots, M_k$ os medoides iniciais e $O_1, \ldots, O_p$ os objetos não medoides. Uma das implementações mais simples do K-medoid é o algoritmo PAM, descrito em [2]. PAM começa com uma seleção aleatória de k objetos. Então, a cada passo, uma troca entre um objeto selecionado M_j e um objeto não selecionado O_i é realizada, desde que essa troca resulte num melhoramento da qualidade do agrupamento. Em especial, para calcular o efeito da troca de M_j por O_i, PAM calcula o custo C_{mij} para todos os objetos não representantes O_m. Dependendo de qual dos casos seguintes O_m corresponde, C_{mij} é definido por uma das equações abaixo (Raymound e Han, 1994):

Caso 1 O_m está no grupo de M_j e com a substituição de M_j por O_i, O_m fica mais próximo de um outro medoide M_{j2}, logo tem-se:

$$C_{mij} = d(O_m, M_{j2}) - d(O_m, M_j) \tag{2.12}$$

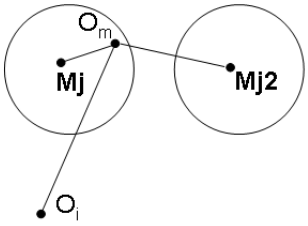


Figura 2.8: Caso 1: O_m iria para o grupo de M_{j2}

Note que C_{mij} será sempre positivo, o que indica um aumento no custo proveniente da troca de O_i com M_j.

Caso 2 O_m está no grupo de M_j e com a substituição de M_j por O_i, O_m fica mais próximo de O_i, tem-se então:

$$C_{mij} = d(O_m, O_i) - d(O_m, M_j) \tag{2.13}$$

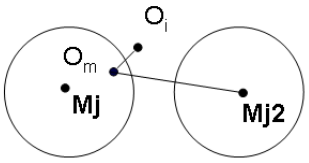


Figura 2.9: Caso 2: O_m iria para o grupo de O_i

Neste caso, C_{mij} poderá ser positivo ou negativo.

Caso 3 O_m não está no grupo de M_j (está no grupo de M_{j2}) e com a substituição de M_j por O_i, O_m continua no grupo de M_{j2} (não muda de grupo), assim tem-se:

$$C_{mij} = d(O_m, M_{j2}) - d(O_m, M_{j2}) = 0 \tag{2.14}$$

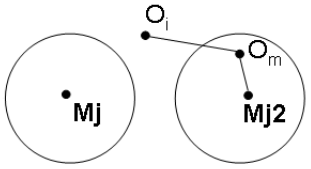


Figura 2.10: Caso 3: O_m continua no mesmo grupo

Caso 4 O_m não está no grupo de M_j (está no grupo de M_{j2}) e com a substituição de M_j por O_i, O_m vai para o grupo de O_i, logo:

$$C_{mij} = d(O_m, O_i) - d(O_m, M_{j2}) \tag{2.15}$$

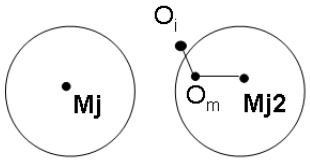


Figura 2.11: Caso 4: O_m iria para o grupo de O_i

Neste caso, C_{mij} sempre será negativo.

Combinando os quatro casos acima, o custo total que representa a troca de M_j com O_i é dado por:

$$CT_{ij} = \sum_{m=1}^{p} C_{mij} \tag{2.16}$$

Algoritmo 2: Algoritmo PAM (Partitioning Around Medoids)

Entrada:

- k: Número de grupos
- D: Base de dados contendo n objetos

Saída: Um conjunto de k grupos

1. Selecione k objetos aleatoriamente.
2. Para cada objeto O_i (O_i não medoide) e cada medoide M_j, calcula-se o custo, CT_{ij}, de trocar M_j por O_i (O_i seria um novo medoide no lugar de M_j)
3. Seleciona-se o par (M_j, O_i) que corresponde ao mínimo CT_{ij}.
   - Se este mínimo é negativo então substitui-se M_j por O_i e volta-se ao passo 2.
   - Se este mínimo é positivo, vai para o passo 4.
4. Varre o banco de dados e distribui os objetos entre os k grupos cujos representantes são os k medoides encontrados no passo 3.

É importante notar no passo 3 do algoritmo 2 que, se o custo total mínimo é negativo, significa que existe uma maneira de se substituir um medoide por outro objeto

de modo a diminuir a soma das distâncias entre cada objeto e seu medoide correspondente. Já se o custo total mínimo é positivo, significa que não há possibilidade de se modificar os medoides atuais de modo a diminuir a soma das distâncias. Logo, neste ponto, os medoides convergiram.

Complexidade de Tempo

PAM funciona satisfatoriamente para pequenos conjuntos de dados (em torno de 100 objetos e 5 grupos), porém é ineficiente para grandes volumes de dados, dado a sua alta complexidade de tempo.

No passo 2 do algoritmo 2, existem $k(n-k)$ pares de M_j, O_i e para cada par é preciso computar C_{mij}, considerando todos os objetos não medoides O_m, logo a complexidade de cada **iteração** é $O(k(n-k)^2)$. Assim, fica óbvio que PAM se torna extremamente custoso para grandes valores de n e k, garantindo que PAM não escala bem para grandes bancos de dados.

Comparando K-means com K-medoid

Segundo Han e Kamber (2006) o K-medoid é mais robusto que o K-means na presença de ruídos e *outliers* porque um medoide é menos influenciado por um *outlier* ou valor extremo do que uma média. Entretanto, seu processamento é mais custoso do que o método K-means. Ambos os métodos necessitam que o usuário especifique o número de grupos, k.

2.3.3 CLARA (Clustering LARge Applications)

Para lidar com grandes bases de dados, um método baseado em amostragem, chamado CLARA pode ser usado, ele é uma variante do método PAM.

A idéia usada no CLARA é a seguinte: Em vez de levar todo o conjunto de dados em consideração, uma pequena porção dos dados é escolhida para representá-los. medoides são escolhidos da amostra utilizando o PAM. Se a amostra for escolhida de uma maneira aleatória justa, ela deverá representar bem a base de dados original. Os medoides escolhidos serão geralmente próximos daqueles que seriam escolhidos utilizando

a totalidade da base de dados. CLARA faz várias amostragens da base de dados, aplica PAM em cada amostra, e retorna o melhor agrupamento como saída. A complexidade de cada iteração agora se torna $O(ks^2 + k(n-k))$, onde s é o tamanho da amostra, k o número de grupos e n o número total de objetos. A eficácia de CLARA depende do tamanho da amostra. Possui uma performance satisfatória para bancos de dados em torno de 1000 objetos e 10 grupos.

2.4 Métodos de Agrupamento Baseado em Modelos

Métodos de agrupamento baseado em modelos tentam otimizar o ajuste entre os dados fornecidos e algum modelo matemático. Tais métodos são geralmente baseados na hipótese que os dados são gerados por uma mistura de distribuições probabilísticas subjacentes.

2.4.1 O Método de Maximização de Expectativas (EM)

Na prática, cada grupo pode ser representado matematicamente por uma distribuição probabilística paramétrica. A totalidade dos dados é uma mistura dessas distribuições, onde cada distribuição individual é tipicamente referida como distribuição componente. Pode-se assim, agrupar os dados usando um modelo de densidade de mistura finita de k distribuições probabilísticas, onde cada distribuição representa um grupo. O problema é estimar os parâmetros das distribuições probabilísticas que melhor se adequam aos dados.

O algoritmo EM é um algoritmo de refinamento iterativo que pode ser usado para encontrar os parâmetros estimados. Pode ser considerado uma extensão do paradigma do K-means, o qual atribui objetos ao grupo que é mais similar, com base na média do grupo. Em vez de atribuir cada objeto a um grupo exclusivo, EM atribui cada objeto para os grupos de acordo com um peso representando a probabilidade do objeto pertencer a tais grupos (figura 2.12). Em outras palavras, não existem limites estritos entre os grupos. Assim, novas médias são computadas baseando-se nas medidas ponderadas. (Han e Kamber, 2006)

O EM inicia-se com uma estimativa inicial dos parâmetros do modelo de mistura (coletivamente referido como vetor de parâmetros). Então, reavalia iterativamente os

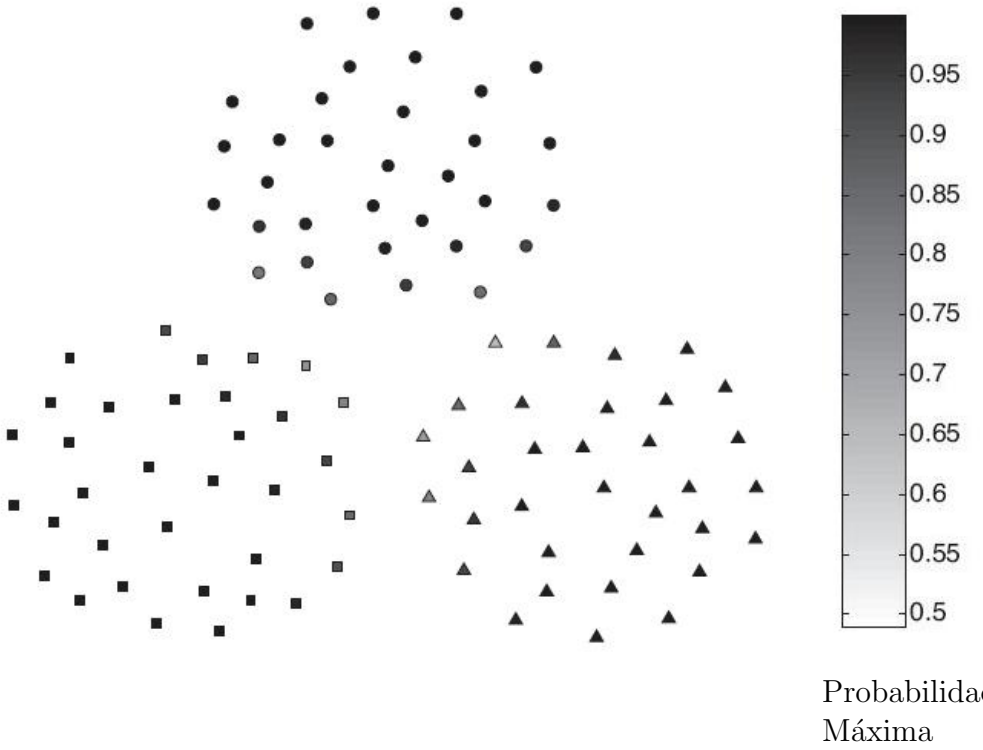


Figura 2.12: Agrupamento EM de um conjunto de pontos bidimensionais com três grupos (Tan et al, 2009)

objetos de acordo com a densidade de mistura produzida pelo vetor de parâmetros. Os objetos reavaliados são então utilizados para atualizar os parâmetros estimados. A cada objeto é atribuído uma probabilidade de que ele possua um certo grupo de valores de atributos, dado que ele pertença a um determinado grupo. O algoritmo é descrito em 3.

Pontos Fortes e Fracos

Segundo Tan et al (2009) modelos de misturas são mais gerais do que K-means porque podem usar distribuições de diversos tipos. Como consequência, modelos misturados (baseados em distribuições Gaussianas) podem encontrar grupos de tamanhos diferentes, densidades diferentes (figura 2.13) e formatos elípticos. A figura 2.14 demonstra como o EM é efetivo ao encontrar grupos elípticos enquanto K-means não consegue encontrar grupos adequados.

Como desvantagem, não é prático para modelos com grande número de componentes e não funciona bem quando os grupos contêm apenas alguns pontos de dados ou se os pontos de dados forem quase co-lineares.

Algoritmo 3: Algoritmo EM

Entrada:

- k: Número de grupos
- D: Base de dados contendo n objetos

Saída: Um conjunto de k grupos

1. Selecione um conjunto inicial de parâmetros de modelos.(Assim como no K-means, isto pode ser feito aleatoriamente, em uma diversidade de formas.)
2. **repita**
3. **Etapa da Expectativa** Para cada objeto, calcule a probabilidade de que o objeto pertença a cada distribuição.
4. **Etapa de Maximização** Dadas as probabilidades da etapa de expectativa, encontre as novas expectativas dos parâmetros que maximizem a probabilidade esperada.
5. **até que** Os parâmetros não mudem.

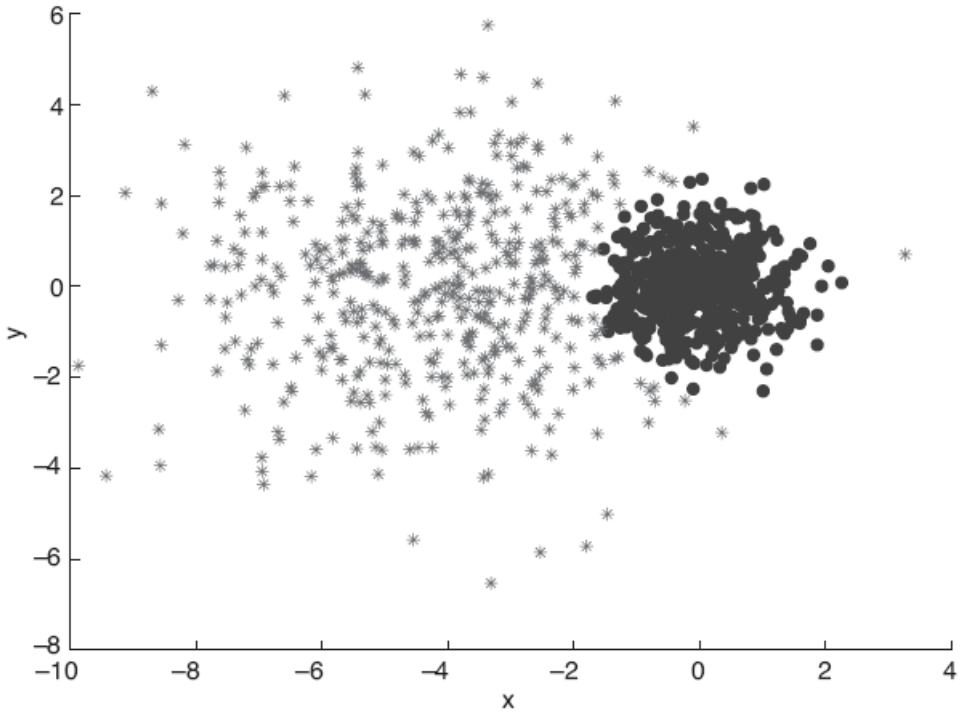
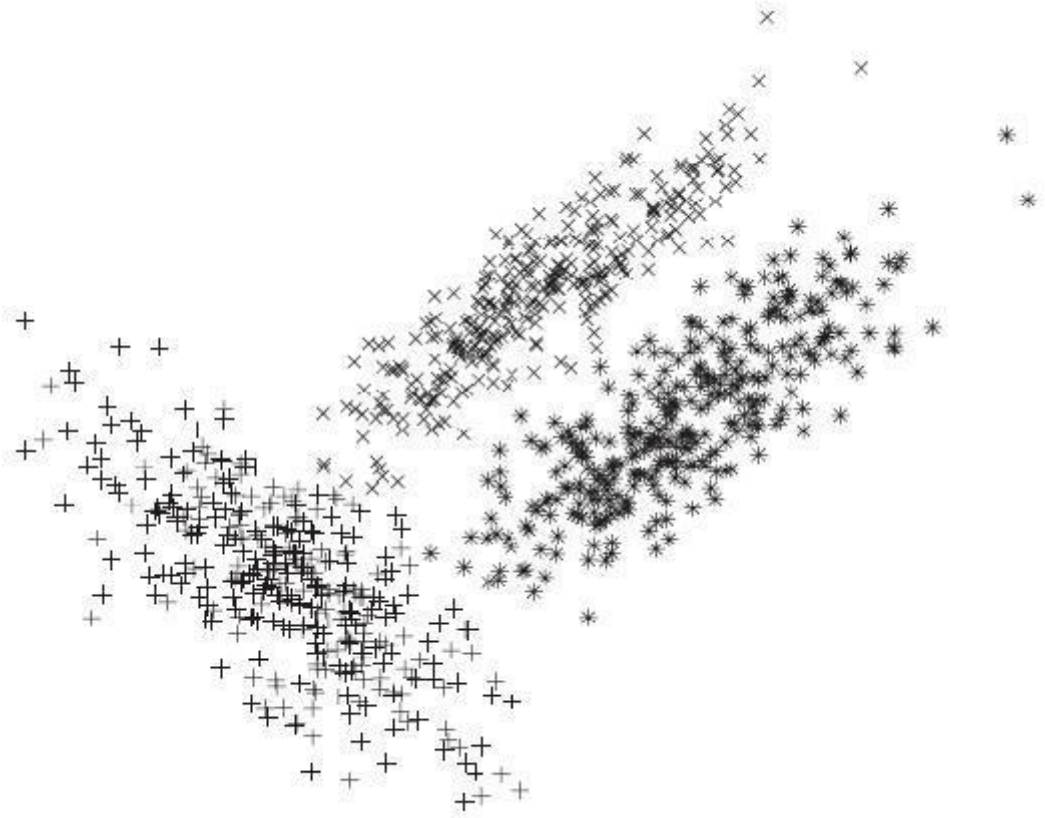
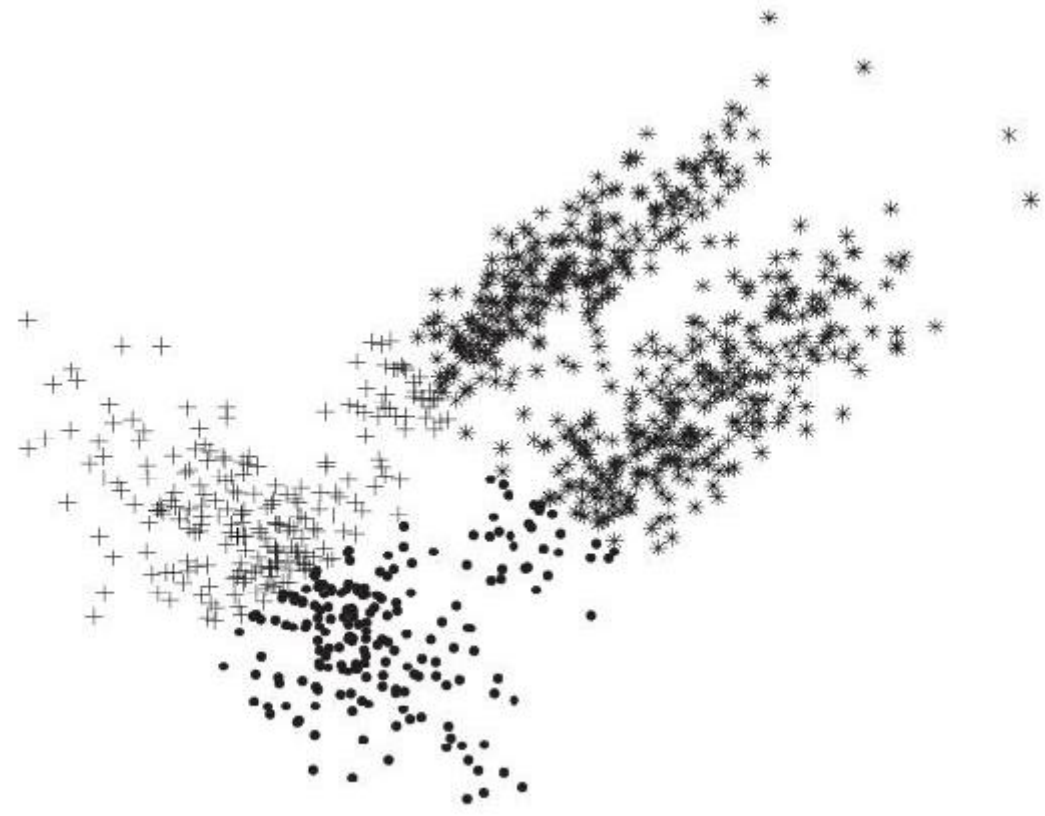


Figura 2.13: Agrupamento EM com um conjunto de pontos bidimensionais com dois grupos com densidades diferentes (Tan et al, 2009)



(a) Grupos produzidos pelo agrupamento EM.



(b) Grupos produzidos por agrupamento K-means.

Figura 2.14: Modelo EM e agrupamento K-means de um conjunto de pontos bidimensionais. (Tan et al, 2009)

3 Implementações

Nesta seção serão descritos as seis base de dados utilizadas, os dois métodos implementados e o software de mineração de dados utilizado, o WEKA.

3.1 Bases de Dados

Foram utilizadas seis bases de dados tradicionais, descritas abaixo:

Íris Provavelmente a base de dados mais conhecida na mineração de dados. Contém 150 amostras, sem valores faltando, divididas em 3 classes, com 50 amostras cada, correspondendo a 33,3% de amostras em cada classe, que referem-se à flores da planta íris. Uma dessas classes é linearmente separável das outras duas. Possui 4 atributos númericos intervalares e uma classe, assim definidos:

1. Comprimento da sépala em centímetros
2. Largura da sépala em centímetros
3. Comprimento da pétala em centímetros
4. Largura da pétala em centímetros
5. Classe:
    - Íris Setosa
    - Íris Versicolour
    - Íris Virginica

Mushroom Esta base de dados contém 5644 amostras de variados cogumelos, todas completas, divididas em 2 classes que distinguem se o cogumelo é comestível ou não. 3488 amostras são da classe comestível, correspondendo a 61,8% da base total e 2156 ou 38,2% são da classe venenoso. Possui 22 atributos nominais, porém foi feita um pré-processamento na base de dados e notou-se que o atributo 16 tinha o

mesmo valor para todas as amostras, portanto o mesmo foi retirado para melhora dos resultados dos algoritmos. Os atributos estão descritos abaixo:

1. Formato do chapéu do cogumelo
2. Textura do chapéu
3. Cor do chapéu
4. Feridas
5. Odor
6. Ligação das lâminas
7. Espaço das lâminas
8. Tamanho das lâminas
9. Cor das lâminas
10. Formato do talo
11. Formato da raiz do talo
12. Textura do talo acima do anel
13. Textura do talo abaixo do anel
14. Cor do talo acima do anel
15. Cor do talo abaixo do anel
16. Tipo do véu (retirado)
17. Cor do véu
18. Quantidade de anel
19. Tipo do anel
20. Cor da impressão dos esporos
21. Tamanho da população
22. Habitat

Sonar Possui 208 amostras completas divididas em 2 classes que distinguem se o sonar de um radar encontrou metal ou pedra no seu caminho. 111 amostras (53,4%) indicam metal e 97 amostras ou 46,6% indicam pedra. Possui 60 atributos intervalares que indicam os dados retornados pelo sonar.

Vowel Possui 990 amostras completas divididas em 11 classes que reconhecem fonemas de vogais orais do inglês britânico que possuem o som estável. Cada classe possui 90 amostras ou 9,1%. Possui 10 atributos intervalares.

Wine Possui 178 amostras completas divididas em 3 classes que representam qual o tipo de vinho da amostra. 59 amostras ou 33,2% representam o vinho 1. 71 amostras ou 39,9% representam o vinho 2 e 48 ou 26,9% representam o vinho 3. Possui 13 atributos intervalares que representam as propriedades do vinho, conforme descritos abaixo:

1. Álcool
2. Ácido málico
3. Pó
4. Alcalinidade do pó
5. Magnésio
6. Fenóis totais
7. Flavonoides
8. Fenóis não flavonoides
9. Proantocianidinas
10. Intensidade da cor
11. Tonalidade
12. OD280/OD315 de vinhos diluídos
13. Prolina

WNBA Contém 120 amostras completas dividas em 3 classes que representam a posição de um jogador de basquete da NBA, das quais, 53 amostras ou 44,16% representam os armadores, 47 ou 39,16% são os alas e 20 ou 16,66% são os pivôs. É importante ressaltar que armadores são geralmente menores que os alas que são menores que os pivôs. Contém 2 atributos intervalares, descritos abaixo:

1. Altura em pés
2. Peso em libras

3.2 Métodos Implementados

Foram implementados para este trabalho dois métodos, o K-means e o algoritmo PAM do K-medoid, ambos utilizando a linguagem C.

3.2.1 Representação da Base de Dados

Em ambos os métodos, o arquivo de entrada de dados, deve ser em formato de texto, *.txt, com os dados possuindo a seguinte estrutura:

- Cada amostra deve estar representada em uma linha.
- A classe da amostra deve estar representada na primeira coluna.
- Cada atributo deve ser separado do seguinte através de um espaço tabulado ou por uma vírgula.
- Valores decimais devem estar separados por ponto.

3.2.2 Inicialização

Antes de executar os algoritmos, é necessário informar:

- A quantidade de grupos a ser utilizada (k).
- A quantidade de amostras na base de dados.
- A quantidade de classes.
- A quantidade de atributos, incluindo a classe real.

3.2.3 Opcionais

É possível executar a base de dados original, ou seja, sem executar qualquer padronização, ou padronizar os dados por escore-z ou normalização linear.

Tanto o escore-z, como a normalização linear foram implementados conforme descrito na subseção 2.1.1.

Também é possível misturar as amostras da base de dados de forma aleatória, antes de submetê-la ao algoritmo.

3.2.4 Outras informações

Em ambos os métodos, os centroides (no caso do K-means) ou objetos representantes (no caso do K-medoid) iniciais, são escolhidos simplesmente de forma aleatória. Portanto, não são realizados cálculos mais complexos para sua escolha.

Por motivo de padronização, a medida utilizada nos testes para calcular as distâncias entre os objetos foi a distância Euclidiana, embora os programas permitam utilizar a distância Manhatam, bastando alterar no código a constante POTENCIA para 1 em vez de 2.

Ao final da execução de cada algoritmo, é mostrado a matriz de confusão para a base de dados, que mostra uma comparação da classe real com a classe encontrada pelos algoritmos. Também é calculada a precisão, que é o quanto o algoritmo acertou, além do número de iterações necessárias para a convergência.

3.2.5 K-Means

O algoritmo implementado utiliza uma matriz de dados para o armazenamento dos atributos, da classe real (localizada na primeira coluna da matriz) e reserva a última coluna da matriz para alocar a classe encontrada pelo algoritmo para cada amostra.

Para a atribuição de uma amostra (ou objeto) a um grupo, é calculada a distância euclidiana do objeto para cada centroide, depois é atribuído ao objeto o índice do centroide mais próximo do mesmo, armazenando esse índice na última coluna da linha da amostra. Esse cálculo é repetido a cada iteração do algoritmo.

3.2.6 K-Medoid

O algoritmo PAM também utiliza uma matriz de dados para armazenar os atributos, a classe real na primeira coluna e a classe encontrada na última coluna.

Ao final de cada iteração, o programa imprime o menor custo total (CT_{ij}) encontrado, que, quando negativo, indica que o algoritmo ainda não convergiu e então haverá a troca do objeto representante atual pelo objeto que melhora os grupos. Quando nenhuma combinação possuir o custo total negativo, a convergência do algoritmo é obtida, com os medoides (os objetos mais centrais de cada grupo) definidos.

3.3 WEKA

O software WEKA (figura 3.1) (*Waikato Environment for Knowledge Analysis*) foi desenvolvido em 1993, usando Java, pela Universidade de Waikato, Nova Zelândia. O Weka encontra-se licenciado pela *General Public License* sendo portanto possível estudar e alterar seu código fonte.

O sistema contém uma série de heurísticas para mineração de dados, entre elas os algoritmos K-means (figura 3.2) e EM, que foram utilizados neste trabalho. Apesar de possuir uma interface gráfica amigável e seus algoritmos fornecerem relatórios com dados analíticos e estatísticos do domínio minerado, sabe-se que cada base de dados tem suas especificidades e características próprias, assim, a falta de conhecimento dos algoritmos prejudica a utilização do WEKA, já que existem dezenas de heurísticas e deve-se ter noção

Figura 3.1: Tela inicial do software WEKA

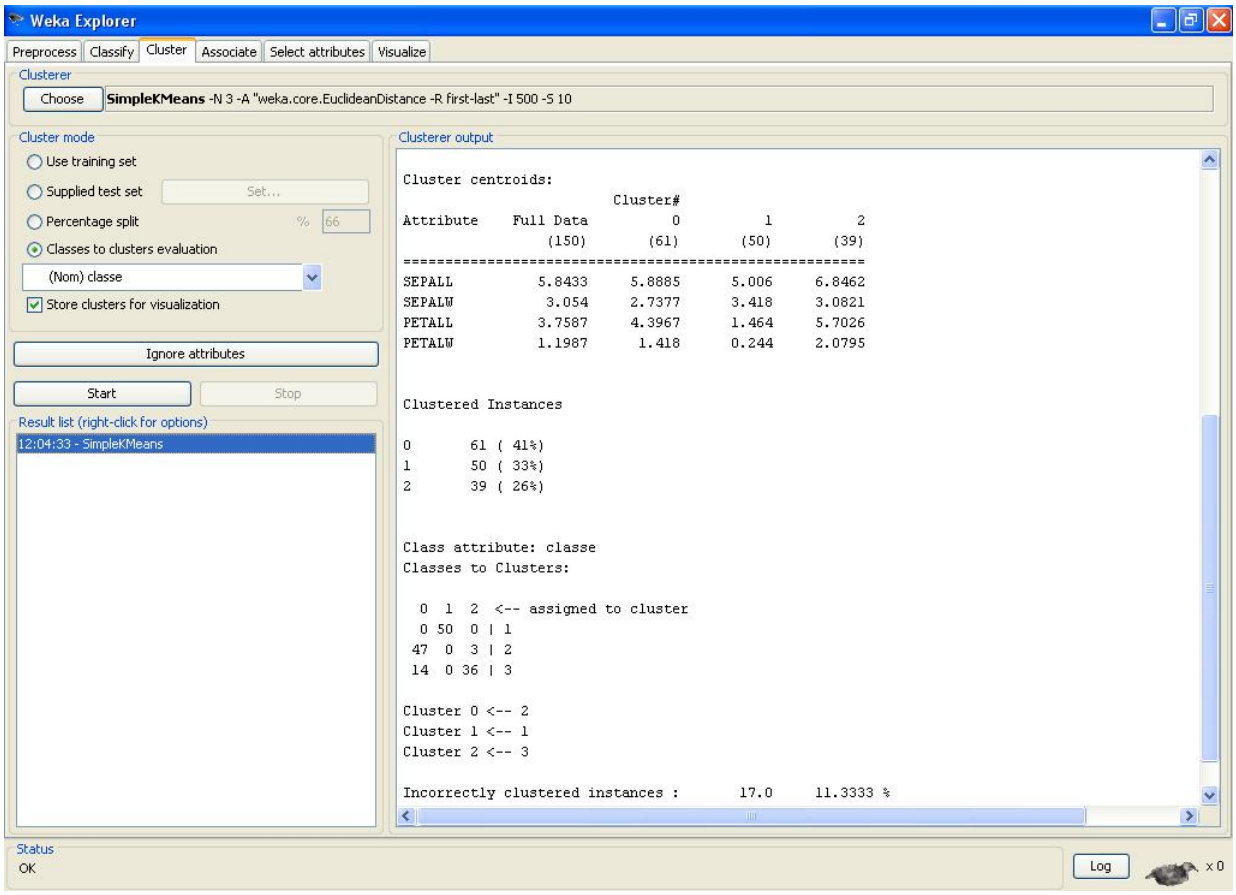


Figura 3.2: Exemplo da tela de execução do K-means para a BD Íris

de qual delas é melhor ou pior para a mineração dos presentes dados.

Para carregar uma base de dados no WEKA é necessário que o arquivo esteja no formato *.arff, no qual define-se quantidade, nomes e tipos dos atributos, nome e quantidade de classes pré-definidas, e cada amostra em uma linha, cujos atributos são separados por vírgula e com a classe real na última coluna. Um exemplo do arquivo de

entrada do WEKA para a base de dados Íris é representado na figura 3.3.

```
% 1. Title: Iris database

@RELATION FlorIris

@ATTRIBUTE SEPALL REAL
@ATTRIBUTE SEPALW REAL
@ATTRIBUTE PETALL REAL
@ATTRIBUTE PETALW REAL
@ATTRIBUTE classe {1,2,3}
@DATA


5.1,3.5,1.4,0.2,1
4.9,3,1.4,0.2,1
4.7,3.2,1.3,0.2,1
4.6,3.1,1.5,0.2,1
5,3.6,1.4,0.2,1
5.4,3.9,1.7,0.4,1
4.6,3.4,1.4,0.3,1
5,3.4,1.5,0.2,1
4.4,2.9,1.4,0.2,1
4.9,3.1,1.5,0.1,1
5.4,3.7,1.5,0.2,1
4.8,3.4,1.6,0.2,1
4.8,3,1.4,0.1,1
4.3,3,1.1,0.1,1
5.8,4,1.2,0.2,1
5.7,4.4,1.5,0.4,1
5.4,3.9,1.3,0.4,1
5.1,3.5,1.4,0.3,1
5.7,3.8,1.7,0.3,1
5.1,3.8,1.5,0.3,1
5.4,3.4,1.7,0.2,1
5.1,3.7,1.5,0.4,1
4.6,3.6,1,0.2,1
5.1,3.3,1.7,0.5,1
4.8,3.4,1.9,0.2,1
5,3,1.6,0.2,1
5,3.4,1.6,0.4,1
5.2,3.5,1.5,0.2,1
```

Figura 3.3: Exemplo de arquivo *.arff para Weka

A versão do WEKA utilizada neste trabalho foi a versão 3.6.4.

4 Resultados Experimentais

Neste trabalho foi realizado um estudo comparativo entre três métodos de agrupamento de modo a verificar o quanto a análise de agrupamento é bastante dependente da base de dados em questão. Observa-se nos testes que um mesmo método pode ter ótimos resultados para determinados dados e resultados ruins para outros. Para isso foi preciso utilizar bases de dados com grupos conhecidos previamente, de forma a viabilizar essa comparação. Também é objetivo do trabalho mostrar a eficiência das implementações realizadas, como o K-means e o K-medoid, tendo inclusive o K-means um mesmo algoritmo implementado pelo WEKA para comparação.

Conforme descrito na subseção 2.3.1, os algoritmos de agrupamento particionais são bastante suscetíveis aos objetos escolhidos como centros iniciais dos grupos. Assim, os algoritmos apresentados neste trabalho foram executados dezenas de vezes, de forma que os resultados apresentados são somente o de melhor precisão para cada situação.

As seis bases de dados foram testadas em todos os métodos (implementação próprias para o K-means e o K-medoid e implementação do WEKA para o K-means e o EM) com seus dados originais, padronizados por normalização linear e por escore-z (Tabelas 4.1, 4.2, 4.3).

Como todos os métodos utilizados são particionais, os resultados foram bastante parecidos. Isso ocorreu principalmente porque todos os métodos produzem agrupamentos globulares. O EM se diferencia dos demais por produzir grupos de densidade e tamanho diferentes. Pode-se deduzir portanto que as bases de dados escolhidas não possuem grupos com tamanho ou densidade muito diferente, pois, caso contrário, os resultados do algoritmo EM seriam significativamente melhor.

Sabe-se que o K-medoid tem uma maior tolerância a valores extremos do que o K-means, por causa do uso de objetos reais como centros dos grupos. Assim, também pode-se concluir que nenhuma das bases de dados clássicas utilizadas possui *outliers*.

Tabela 4.1: Comparativo dos métodos - BD´s originais

Base de Dados Originais				
Identificação	Acerto Percentual (máximo encontrado)			
	K-means	K-medoid	K-means weka	EM
Iris	89,33	90,67	88,67	90,67
Mushroom	84,41	83,88	84,55	85,26
Sonar	56,25	52,40	56,25	53,85
Vowel	36,67	35,05	34,95	37,17
Wine	70,22	72,47	94,94	97,19
WNBA	70,83	72,50	76,67	75,83

Tabela 4.2: Comparativo dos métodos - BD´s com normalização linear

Base de Dados com Normalização Linear				
Identificação	Acerto Percentual (máximo encontrado)			
	K-means	K-medoid	K-means weka	EM
Iris	88,67	90,00	88,67	90,00
Mushroom	84,55	84,41	84,57	85,26
Sonar	56,25	54,81	56,25	53,85
Vowel	36,67	28,18	36,46	37,78
Wine	96,63	92,70	95,51	96,63
WNBA	76,67	76,67	76,67	75,83

Tabela 4.3: Comparativo dos métodos - BD´s com Escores-z

Base de Dados com escores-z				
Identificação	Acerto Percentual (máximo encontrado)			
	K-means	K-medoid	K-means weka	EM
Iris	83,33	84,67	88,67	90,67
Mushroom	85,97	84,48	84,55	85,26
Sonar	60,58	54,81	56,25	53,85
Vowel	34,95	30,71	34,34	37,88
Wine	97,19	89,89	95,51	97,19
WNBA	77,50	78,33	76,67	75,83

4.1 Íris

Para a base de dados Íris, no algoritmo K-means, as padronizações de dados não obtiveram bons resultados, resultando inclusive numa ligeira piora da precisão dos acertos dos dados originais (89.33%) em relação à normalização linear (88,67%) e uma queda um pouco mais acentuada para o escore-z (83,33%) (Tabela 4.4).

Tabela 4.4: K-Means - Base de Dados: Íris

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	50	50	100,00	50	100,00	49	98,00
Classe 2	50	48	96,00	47	94,00	38	76,00
Classe 3	50	36	72,00	36	72,00	38	76,00
Total	150	134	89,33	133	88,67	125	83,33

O algoritmo K-medoid seguiu a mesma tendência (originais: 90,67%; Norm. Linear: 90%; Escore-z: 84,67%), porém com resultados ligeiramente melhores ao algoritmo K-means (Tabela 4.5).

Tabela 4.5: K-Medoid - Base de Dados : Íris

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	50	50	100,00	50	100,00	49	98,00
Classe 2	50	37	74,00	49	98,00	35	70,00
Classe 3	50	49	98,00	36	72,00	43	86,00
Total	150	136	90,67	135	90,00	127	84,67

O algoritmo K-means do WEKA mostrou-se invariável às padronizações, apresentando o mesmo resultado para todos os três tipos de entradas, 88,67% (Tabela 4.6), o que pode indicar um pré-processamento dos dados realizado pelo próprio software. O resultado do K-means WEKA mostrou-se pior que o melhor resultado do K-means.

O algoritmo EM também mostrou-se pouco variável às padronizações, somente errando uma amostra a mais na normalização linear (Tabela 4.7), tendo como melhor resultado, a execução com dados originais e com escore-z, 90,67%.

Tabela 4.6: K-Means WEKA - Base de Dados: Íris

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	50	50	100,00	50	100,00	50	100,00
Classe 2	50	47	94,00	47	94,00	47	94,00
Classe 3	50	36	72,00	36	72,00	36	72,00
Total	150	133	88,67	133	88,67	133	88,67

Tabela 4.7: Maximização de Expectativas (EM) Weka - Base de Dados: Íris

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	50	50	100,00	50	100,00	50	100,00
Classe 2	50	50	100,00	49	98,00	50	100,00
Classe 3	50	36	72,00	36	72,00	36	72,00
Total	150	136	90,67	135	90,00	136	90,67

Entre todos os resultados, os algoritmos com melhor retorno para a base de dados Íris foram o K-medoid com dados originais e o EM com dados originais e com escore-z, com 90,67% de precisão cada.

4.2 Mushroom

Para a base de dados Mushroom, no algoritmo K-means a padronização de dados mostrou uma ligeira melhora nos resultados. Com os dados originais, obteve 84,41% de acertos, melhorando para 84,55% com normalização linear e um resultado um pouco melhor com escore-z, de 85,97% de precisão (Tabela 4.8).

Tabela 4.8: K-Means - Base de Dados: Mushroom

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	3488	3424	98,17	3432	98,39	3440	98,62
Classe 2	2156	1340	62,15	1340	62,15	1412	65,49
Total	5644	4764	84,41	4772	84,55	4852	85,97

O algoritmo K-medoid, seguiu a mesma tendência crescente com as padronizações (originais: 83,88%; Norm. Linear: 84,41%; Escore-z: 84,48%) (Tabela 4.9). É importante ressaltar que o fato de o algoritmo PAM ser bastante custoso reflete nesse resultado. Como

a base de dados Mushroom possui 5644 amostras, podemos observar pelas tabelas A.4, A.5 e A.6 (Anexo A) que o tempo médio de execução do algoritmo foi de aproximadamente 27 minutos, comprovando que o algoritmo PAM não é viável para grandes bases de dados.

Tabela 4.9: K-Medoid - Base de Dados: Mushroom

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	3488	3388	97,13	3424	98,17	3428	98,28
Classe 2	2156	1346	62,43	1340	62,15	1340	62,15
Total	5644	4734	83,88	4764	84,41	4768	84,48

O algoritmo K-means do WEKA mostrou-se mais uma vez invariável às padronizações, tendo dessa vez apenas um acerto a mais com a normalização linear dos dados, 84,57% ou 4773 amostras corretas, contra 84,55% ou 4772 amostras da base de dados original e com escore-z (Tabela 4.10). Comparando a implementação do pacote de software WEKA com a implementação específica deste trabalho, mais uma vez temos um resultado ligeiramente melhor da implementação deste trabalho.

Tabela 4.10: K-Means Weka - Base de Dados: Mushroom

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	3488	3432	98,39	3433	98,42	3432	98,39
Classe 2	2156	1340	62,15	1340	62,15	1340	62,15
Total	5644	4772	84,55	4773	84,57	4772	84,55

O algoritmo EM obteve o mesmo resultado para todas as padronizações, 85,26% de acerto (Tabela 4.11).

Tabela 4.11: Maximização de Expectativas (EM) Weka - Base de Dados: Mushroom

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	3488	3472	99,54	3472	99,54	3472	99,54
Classe 2	2156	1340	62,15	1340	62,15	1340	62,15
Total	5644	4812	85,26	4812	85,26	4812	85,26

Entre todos os resultados, o melhor encontrado foi com o algoritmo K-means deste trabalho, que com os dados padronizados através do escore-z, obteve 85,97% de acerto.

4.3 Sonar

Na base de dados Sonar, no algoritmo K-means, as versões original e com normalização linear, obtiveram 56,25% de acertos, já os dados com escore-z tiveram um melhor resultado, com 60,58% de acerto (Tabela 4.12).

Tabela 4.12: K-Means - Base de Dados: Sonar

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	111	58	52,25	59	53,15	45	40,54
Classe 2	97	59	60,82	58	59,79	81	83,51
Total	208	117	56,25	117	56,25	126	60,58

O algoritmo K-medoid obteve piores resultados. Atingiu 52,40% de acerto com a base de dados original e 54,81% de acerto nas bases de dados com normalização linear e escore-z (Tabela 4.13).

Tabela 4.13: K-Medoid - Base de Dados: Sonar

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	111	46	41,44	54	48,65	54	48,65
Classe 2	97	63	64,95	60	61,86	60	61,86
Total	208	109	52,40	114	54,81	114	54,81

O algoritmo K-means do WEKA obteve 56,25% de acerto em todas as bases de dados. Sendo portanto pior que o melhor resultado do seu equivalente implementado neste trabalho, que obteve 60,58% com escore-z (Tabela 4.14).

O algoritmo EM alcançou o pior resultado de todos os métodos mas ainda assim bastante próximo dos mesmos. Teve 53,85% de acertos (Tabela 4.15).

Entre todos os resultados, o melhor obtido foi com o K-means implementado para este trabalho, que padronizado por escore-z obteve 60,58% de acertos. Esse resultado não

Tabela 4.14: K-Means Weka - Base de Dados: Sonar

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	111	58	52,25	58	52,25	58	52,25
Classe 2	97	59	60,82	59	60,82	59	60,82
Total	208	117	56,25	117	56,25	117	56,25

Tabela 4.15: Maximização de Expectativas (EM) Weka - Base de Dados: Sonar

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	111	56	50,45	56	50,45	56	50,45
Classe 2	97	56	57,73	56	57,73	56	57,73
Total	208	112	53,85	112	53,85	112	53,85

garante a certeza de conseguir um bom agrupamento, através dos métodos testados para agrupar a base de dados Sonar.

4.4 Vowel

Para a base de dados Vowel, no algoritmo K-means, a padronização não resultou em melhores resultados. Com dados originais e normalização linear o algoritmo obteve 36,67% de acertos, já com escore-z o resultado foi pior, 34,95% (Tabela 4.16).

Tabela 4.16: K-Means - Base de Dados: Vowel

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	90	51	56,67	44	48,89	37	41,11
Classe 2	90	24	26,67	34	37,78	33	36,67
Classe 3	90	29	32,22	36	40,00	35	38,89
Classe 4	90	60	66,67	53	58,89	52	57,78
Classe 5	90	46	51,11	41	45,56	35	38,89
Classe 6	90	30	33,33	31	34,44	29	32,22
Classe 7	90	37	41,11	18	20,00	20	22,22
Classe 8	90	16	17,78	34	37,78	42	46,67
Classe 9	90	14	15,56	24	26,67	16	17,78
Classe 10	90	24	26,67	29	32,22	24	26,67
Classe 11	90	32	35,56	19	21,11	23	25,56
Total	990	363	36,67	363	36,67	346	34,95

No algoritmo K-medoid, o melhor resultado foi com a base de dados original, com 35,05% de acerto. As duas padronizações, normalização linear e escore-z, obtiveram resultados bem abaixo que os dados originais, respectivamente 28,18% e 30,71% (Tabela 4.17). Assim como em Mushroom, a base de dados Vowel é bastante extensa, contando com 990 amostras e 10 atributos, e além disso ainda possui 11 classes, o que leva a um aumento no tempo de convergência para os medoides, como pode-se observar nas tabelas A.4, A.5 e A.6 (Anexo A), nas quais obteve-se 13,6 iterações em média, o que resultou num tempo médio de execução de 33 minutos.

Tabela 4.17: K-Medoid - Base de Dados: Vowel

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	90	34	37,78	28	31,11	33	36,67
Classe 2	90	16	17,78	23	25,56	23	25,56
Classe 3	90	48	53,33	32	35,56	25	27,78
Classe 4	90	0	0,00	65	72,22	44	48,89
Classe 5	90	27	30,00	0	0,00	28	31,11
Classe 6	90	58	64,44	22	24,44	9	10,00
Classe 7	90	18	20,00	12	13,33	16	17,78
Classe 8	90	77	85,56	29	32,22	49	54,44
Classe 9	90	17	18,89	27	30,00	24	26,67
Classe 10	90	26	28,89	23	25,56	17	18,89
Classe 11	90	26	28,89	18	20,00	36	40,00
Total	990	347	35,05	279	28,18	304	30,71

O algoritmo K-means do WEKA obteve resultados bastante variáveis. A base de dados original teve 34,95% de acertos. A normalização linear dos dados gerou um acerto maior, de 36,46%. Já o escore-z foi pior que os dados originais, 34,34% (Tabela 4.18).

O algoritmo EM foi bastante regular e obteve os melhores resultados. Com a base de dados original teve 37,17% de acerto, com normalização linear 37,78% e com escore-z o melhor resultado: 37,88% (Tabela 4.19).

O algoritmo EM foi o melhor para a base de dados Vowel de forma absoluta. Em todas as variações da base de dados ele obteve o melhor resultado, sendo com escore-z o melhor absoluto, com 37,88%, apesar disso os resultados ainda são bastante ruins, caracterizando a base de dados Vowel como de difícil análise.

Tabela 4.18: K-Means Weka - Base de Dados: Vowel

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	90	35	38,89	32	35,56	39	43,33
Classe 2	90	23	25,56	23	25,56	32	35,56
Classe 3	90	41	45,56	41	45,56	12	13,33
Classe 4	90	53	58,89	53	58,89	52	57,78
Classe 5	90	27	30,00	42	46,67	39	43,33
Classe 6	90	31	34,44	31	34,44	24	26,67
Classe 7	90	35	38,89	35	38,89	31	34,44
Classe 8	90	48	53,33	46	51,11	46	51,11
Classe 9	90	3	3,33	9	10,00	12	13,33
Classe 10	90	31	34,44	30	33,33	24	26,67
Classe 11	90	19	21,11	19	21,11	29	32,22
Total	990	346	34,95	361	36,46	340	34,34

Tabela 4.19: Maximização de Expectativas (EM) Weka - Base de Dados: Vowel

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	90	35	38,89	35	38,89	34	37,78
Classe 2	90	21	23,33	21	23,33	21	23,33
Classe 3	90	36	40,00	36	40,00	36	40,00
Classe 4	90	44	48,89	59	65,56	62	68,89
Classe 5	90	56	62,22	49	54,44	50	55,56
Classe 6	90	30	33,33	23	25,56	23	25,56
Classe 7	90	44	48,89	38	42,22	37	41,11
Classe 8	90	48	53,33	48	53,33	48	53,33
Classe 9	90	13	14,44	16	17,78	15	16,67
Classe 10	90	18	20,00	24	26,67	24	26,67
Classe 11	90	23	25,56	25	27,78	25	27,78
Total	990	368	37,17	374	37,78	375	37,88

4.5 Wine

Para a base de dados Wine, no algoritmo K-means, a padronização dos dados funcionou muito bem. Com os dados originais, o resultado foi 70,22% de acerto. Já com a normalização linear e escore-z, foram respectivamente, 96,63% e 97,19%, com 2/3 das classes obtendo 100% de acerto (Tabela 4.20).

No algoritmo K-medoid, observa-se o mesmo ganho de resultado com as padronizações. A base de dados original obteve 72,47% de acerto. As bases padronizadas por normalização linear e escore-z tiveram respectivamente, 92,70% e 89,89% (Tabela 4.21).

Tabela 4.20: K-Means - Base de Dados: Wine

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	59	46	77,97	59	100,00	59	100,00
Classe 2	71	50	70,42	65	91,55	66	92,96
Classe 3	48	29	60,42	48	100,00	48	100,00
Total	178	125	70,22	172	96,63	173	97,19

Esse ganho com a padronização é proveniente do fato da mesma proporcionar a todos os atributos uma mesma escala de medidas, excluindo com isso possíveis distorções geradas por atributos com grande valores nos dados e outros com pequenos valores que atribuem maior peso no resultado aos valores de um determinado atributo. Esse é o caso da base Wine, que possui um atributo com valores na escala de milhares de unidades, como também um atributo com valores na escala decimal.

Tabela 4.21: K-Medoid - Base de Dados: Wine

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	59	48	81,36	59	100,00	59	100,00
Classe 2	71	49	69,01	58	81,69	53	74,65
Classe 3	48	32	66,67	48	100,00	48	100,00
Total	178	129	72,47	165	92,70	160	89,89

O algoritmo K-means do WEKA também obteve ótimos resultados, mas sem grandes variações entre as bases testadas, por motivos já informados. A base de dado original obteve 94,94% de acerto. Com normalização linear e escore-z o resultado foi de 95,51% para ambas (Tabela 4.22).

Tabela 4.22: K-Means Weka - Base de Dados: Wine

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	59	59	100,00	59	100,00	59	100,00
Classe 2	71	62	87,32	63	88,73	63	88,73
Classe 3	48	48	100,00	48	100,00	48	100,00
Total	178	169	94,94	170	95,51	170	95,51

O algoritmo EM obteve 97,19% com base original e com escore-z. E um resultado menor com normalização linear, 96,63%. É importante ressaltar que apesar do excelente

resultado, ao contrário dos métodos particionais, o EM não acertou 100% duas classes, somente uma (Tabela 4.23). Em compensação obteve resultados na classe 2 muito próximos do máximo, enquanto os métodos particionais foram piores em reconhecer essa classe.

Tabela 4.23: Maximização de Expectativas (EM) Weka - Base de Dados: Wine

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	59	57	96,61	56	94,92	57	96,61
Classe 2	71	68	95,77	68	95,77	68	95,77
Classe 3	48	48	100,00	48	100,00	48	100,00
Total	178	173	97,19	172	96,63	173	97,19

Entre todos os resultados, o algoritmo K-means implementado para este trabalho com escore-z, juntamente com o algoritmo EM com dados originais e padronizados por escore-z obtiveram o melhor resultado, com 97,19% de acerto. Atestando a eficácia e confiabilidade desses métodos ao lidar com essa base de dados.

4.6 WNBA

Para a base de dados WNBA, no algoritmo K-means, também observou-se uma melhora nos resultados com as padronizações, porém não tão acentuadas como foi com a base de dados Wine, pois na WNBA um atributo está na casa da unidade e o outro na casa das centenas. Com a base original, obteve-se 70,83% de acerto, com normalização linear e escore-z, respectivamente, 76,67% e 77,50% (Tabela 4.24).

Tabela 4.24: K-Means - Base de Dados: WNBA

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	53	43	81,13	44	83,02	45	84,91
Classe 2	47	35	74,47	36	76,60	37	78,72
Classe 3	20	7	35,00	12	60,00	11	55,00
Total	120	85	70,83	92	76,67	93	77,50

O algoritmo K-medoid, também teve um melhor resultado com as padronizações, apresentando 72,50% de acerto com a base original e 78,33% com a normalização linear e escore-z (Tabela 4.25).

Tabela 4.25: K-Medoid - Base de Dados: WNBA

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	53	45	84,91	48	90,57	48	90,57
Classe 2	47	35	74,47	34	72,34	34	72,34
Classe 3	20	7	35,00	12	60,00	12	60,00
Total	120	87	72,50	94	78,33	94	78,33

O algoritmo K-means do WEKA obteve o mesmo índice de acerto, 76,67% em todas as bases de dados (Tabela 4.26). Já o algoritmo EM atingiu 75,83% (Tabela 4.27) de acerto também em todas as bases de dados.

Tabela 4.26: K-Means Weka - Base de Dados: WNBA

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	53	44	83,02	45	84,91	45	84,91
Classe 2	47	36	76,60	35	74,47	37	78,72
Classe 3	20	12	60,00	12	60,00	10	50,00
Total	120	92	76,67	92	76,67	92	76,67

Tabela 4.27: Maximização de Expectativas (EM) Weka - Base de Dados: WNBA

Acertos por classe							
		BD original		BD com norm. linear		BD com escores-z	
Classes	Amostras	Abs.	Per.	Abs.	Per.	Abs.	Per.
Classe 1	53	45	84,91	45	84,91	45	84,91
Classe 2	47	37	78,72	37	78,72	37	78,72
Classe 3	20	9	45,00	9	45,00	9	45,00
Total	120	91	75,83	91	75,83	91	75,83

O algoritmo K-medoid foi o que obteve o melhor resultado com a base de dados WNBA, atingindo 78,33% com padronização por escore-z e normalização linear. Um resultado satisfatório.

4.7 Conclusão

O algoritmo K-means implementado para este trabalho obteve resultados muito bons em comparação com os demais algoritmos. Das 6 bases de dados testadas, ele foi o melhor em

3 delas, sendo duas delas de forma absoluta, todas com as bases de dados padronizadas com escore-z.

Como ocorreram resultados idênticos, observou-se que o algoritmo K-medoid conseguiu o maior percentual de acertos em duas ocasiões e o algoritmo EM teve o maior percentual em três bases de dados. O algoritmo K-means do WEKA não conseguiu obter o maior percentual de acerto em nenhuma das bases de dados utilizadas, porém apesar disso, seu resultado, assim como todos os outros sempre estiveram próximos entre si.

Entre todos os melhores resultados, sempre esteve presente a padronização por escore-z, provando que sua utilização quase sempre resulta numa melhora dos resultados, já a normalização linear somente esteve presente em um dos melhores resultados, porém seu resultado nunca esteve muito abaixo das bases de dados originais ou com escore-z.

Provou-se que na análise de agrupamento, o tipo de método utilizado depende bastante do problema em questão, pois os mesmos métodos podem obter excelentes resultados, mas também resultados muito ruins.

Podemos atribuir os resultados ruins da base de dados Sonar ao fato da mesma conter 60 atributos e os métodos utilizados neste trabalho não conseguem realizar um bom agrupamento diante de muitos atributos.

No anexo A, encontram-se os resumos em tabelas separadas por métodos, com e sem padronização dos dados. Possui dados resumidos das bases de dados, como: tempo de execução do algoritmo, acertos e quantidade de iterações necessárias, entre outros.

No anexo B, estão expostas todas as matrizes de confusão de cada base de dado em cada método.

5 Considerações Finais

Com o crescente avanço da tecnologia, é cada vez mais comum o armazenamento de imensas quantidades de dados, tornando inviável a análise desses dados sem o uso de alguma ferramenta automática.

Neste trabalho foram apresentados conceitos de mineração de dados e de análise de agrupamento, que surgiram tendo como um de seus objetivos a criação de um meio rápido de analisar grandes quantidades de dados de forma a obter padrões úteis que identificariam oportunidades de negócios, avanços científicos, detecção de fraudes etc.

Também foram desenvolvidas as implementações de dois métodos conhecidos de análise de agrupamento, K-means e K-medoid, e implementações de técnicas de padronização dos dados, que juntos com outras implementações do pacote de software WEKA, foram testados com diversas bases de dados, a fim de mostrar que o resultado da análise de agrupamento é bastante dependente dos tipos de dados a serem explorados e do conhecimento do problema. Por isso, a escolha do método a ser utilizado na análise deve ser feita com bastante cuidado. Ficou também demonstrado que a padronização de dados pode influenciar significativamente para a obtenção de melhores resultados.

5.1 Trabalhos Futuros

Este trabalho é apenas uma parte do que pode ser explorado com a mineração de dados e mais especificamente com a análise de agrupamento. Assim, sugerem-se os seguintes trabalhos futuros:

- Implementação de variações dos métodos apresentados, como exemplo, CLARA.
- Aperfeiçoamento dos algoritmos implementados de forma a poderem lidar melhor com outros tipos de dados.

Referências Bibliográficas

Fayyad, U.; Piatetsky-Shapiro, G. ; Smyth, P. **From Data Mining to Knowledge Discovery in Databases**, volume 17. Al Magazine, 1996.

Hair, J. F.; Anderson, R. E.; Thatam, R. L. ; Black, W. C. **Análise Multivariada de Dados**. quinta. ed., Porto Alegre, RS, Brasil: Bookman Companhia Editora, 2005.

Han, J.; Kamber, M. **Data Mining: Concepts and Techniques**. second. ed., São Francisco, CA, EUA: Morgan Kaufmann Publishers, 2006.

da Motta, C. G. L. **Sistema inteligente para avaliação de riscos em vias de transporte terrestre**. Rio de Janeiro, RJ, Brasil, 2004. Dissertação de Mestrado - COPPE/UFRJ.

Raymound, T. N.; Han, J. Efficient and effective clustering methods for spatial data mining. **VLDB Conference**, 1994.

Rezende, S. O.; Pugliesi, J. B.; Melanda, E. A. ; de Paula, M. F. **Sistemas Inteligentes: Fundamentos e Aplicações**. Barueri, SP, Brasil: Editora Manole Ltda., 2003.

Tan, P.-N.; Steinback, M. ; Kumar, V. **Introdução ao Data Mining: Mineração de Dados**. Rio de Janeiro, RJ, Brasil: Editora Ciência Moderna Ltda., 2009.

A Resumo das Bases de Dados por Métodos

Tabela A.1: Resumo K-Means - BD´s originais

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Íris	4	-	4	3	5	0,7s	150	134	89,33
Mushroom	22	22	-	2	5	0,8s	5644	4764	84,41
Sonar	60	-	60	2	18	0,7s	208	117	56,25
Vowel	10	-	10	11	21	1,1s	990	363	36,67
Wine	13	-	13	3	7	0,8s	178	125	70,22
WNBA	2	-	2	3	6	0,6s	120	85	70,83

Tabela A.2: Resumo K-Means - BD´s com normalização linear

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	7	0,5s	150	133	88,67
Mushroom	22	22	-	2	5	1,1s	5644	4772	84,55
Sonar	60	-	60	2	9	0,7s	208	117	56,25
Vowel	10	-	10	11	20	1,5s	990	363	36,67
Wine	13	-	13	3	7	1,6s	178	172	96,63
WNBA	2	-	2	3	6	0,7s	120	92	76,67

Tabela A.3: Resumo K-Means - BD´s com escores-z

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	13	0,6s	150	125	83,33
Mushroom	22	22	-	2	6	1,6s	5644	4852	85,97
Sonar	60	-	60	2	10	1s	208	126	60,58
Vowel	10	-	10	11	30	1,3s	990	346	34,95
Wine	13	-	13	3	6	1,8s	178	173	97,19
WNBA	2	-	2	3	6	0,5s	120	93	77,50

Tabela A.4: Resumo K-Medoid - BD´s originais

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	3	0,5s	150	136	90,67
Mushroom	22	22	-	2	4	1800s (30 min)	5644	4734	83,88
Sonar	60	-	60	2	2	6,2s	208	109	52,40
Vowel	10	-	10	11	13	2087s (35 min)	990	347	35,05
Wine	13	-	13	3	4	3,8s	178	129	72,47
WNBA	2	-	2	3	3	0,2s	120	87	72,50

Tabela A.5: Resumo K-Medoid - BD´s com normalização linear

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	3	0,5s	150	135	90,00
Mushroom	22	22	-	2	2	1526s (25 min)	5644	4764	84,41
Sonar	60	-	60	2	2	6s	208	114	54,81
Vowel	10	-	10	11	15	1740s (29 min)	990	279	28,18
Wine	13	-	13	3	5	3,4s	178	165	92,70
WNBA	2	-	2	3	6	0,7s	120	92	76,67

Tabela A.6: Resumo K-Medoid - BD´s com escores-z

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	4	0,7s	150	127	84,67
Mushroom	22	22	-	2	2	1485s (25 min)	5644	4768	84,48
Sonar	60	-	60	2	3	7,6s	208	114	54,81
Vowel	10	-	10	11	13	2084s (35 min)	990	304	30,71
Wine	13	-	13	3	5	3,9s	178	160	89,89
WNBA	2	-	2	3	3	0,2s	120	94	78,33

Tabela A.7: Resumo K-Means Weka - BD's originais

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	6	-	150	133	88,67
Mushroom	22	22	-	2	6	-	5644	4772	84,55
Sonar	60	-	60	2	12	-	208	117	56,25
Vowel	10	-	10	11	35	-	990	346	34,95
Wine	13	-	13	3	5	-	178	169	94,94
WNBA	2	-	2	3	8	-	120	92	76,67

Tabela A.8: Resumo K-Means Weka - BD's com normalização linear

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	5	-	150	133	88,67
Mushroom	22	22	-	2	6	-	5644	4773	84,57
Sonar	60	-	60	2	12	-	208	117	56,25
Vowel	10	-	10	11	30	-	990	361	36,46
Wine	13	-	13	3	6	-	178	170	95,51
WNBA	2	-	2	3	3	-	120	92	76,67

Tabela A.9: Resumo K-Means Weka - BD's com escores-z

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	8	-	150	133	88,67
Mushroom	22	22	-	2	5	-	5644	4772	84,55
Sonar	60	-	60	2	12	-	208	117	56,25
Vowel	10	-	10	11	34	-	990	340	34,34
Wine	13	-	13	3	7	-	178	170	95,51
WNBA	2	-	2	3	4	-	120	92	76,67

Tabela A.10: Resumo Maximização de Expectativas(EM) Weka - BD's originais

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	-	-	150	136	90,67
Mushroom	22	22	-	2	-	-	5644	4812	85,26
Sonar	60	-	60	2	-	-	208	112	53,85
Vowel	10	-	10	11	-	-	990	368	37,17
Wine	13	-	13	3	-	-	178	173	97,19
WNBA	2	-	2	3	-	-	120	91	75,83

Tabela A.11: Resumo Maximização de Expectativas(EM) Weka - BD's com normalização linear

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	-	-	150	135	90,00
Mushroom	22	22	-	2	-	-	5644	4812	85,26
Sonar	60	-	60	2	-	-	208	112	53,85
Vowel	10	-	10	11	-	-	990	374	37,78
Wine	13	-	13	3	-	-	178	172	96,63
WNBA	2	-	2	3	-	-	120	91	75,83

Tabela A.12: Resumo Maximização de Expectativas(EM) Weka - BD's com escores-z

Base de Dados									
	Atributos							Acertos (máximo encontrado)	
Identificação	N	ND	NC	Classes	Iterações	Tempo	Amostras	Absoluto	Percentual
Iris	4	-	4	3	-	-	150	136	90,67
Mushroom	22	22	-	2	-	-	5644	4812	85,26
Sonar	60	-	60	2	-	-	208	112	53,85
Vowel	10	-	10	11	-	-	990	375	37,88
Wine	13	-	13	3	-	-	178	173	97,19
WNBA	2	-	2	3	-	-	120	91	75,83

B Matrizes de Confusão

B.1 Base de dados Íris

Tabela B.1: K-Means - BD: Íris original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	48	2	50
	3	0	14	36	50
	Total	50	62	38	150

Tabela B.2: K-Means - BD: Íris com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	47	3	50
	3	0	14	36	50
	Total	50	61	39	150

Tabela B.3: K-Means - Íris com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	49	1	0	50
	2	0	38	12	50
	3	0	12	38	50
	Total	49	51	50	150

Tabela B.4: K-Medoid - BD: Íris original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	37	13	50
	3	0	1	49	50
	Total	50	38	62	150

Tabela B.5: K-Medoid - BD: Íris com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	49	1	50
	3	0	14	36	50
	Total	50	63	37	150

Tabela B.6: K-Medoid - BD: Íris com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	49	1	0	50
	2	0	35	15	50
	3	0	7	43	50
	Total	49	43	58	150

Tabela B.7: K-Means Weka - BD: Íris original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	47	3	50
	3	0	14	36	50
	Total	50	61	39	150

Tabela B.8: K-Means Weka - BD: Íris com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	47	3	50
	3	0	14	36	50
	Total	50	61	39	150

Tabela B.9: K-Means Weka - BD: Íris com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	47	3	50
	3	0	14	36	50
	Total	50	61	39	150

Tabela B.10: Maximização de Expectativas (EM) Weka - BD: Íris original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	50	0	50
	3	0	14	36	50
	Total	50	64	36	150

Tabela B.11: Maximização de Expectativas (EM) Weka - BD: Íris com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	49	1	50
	3	0	14	36	50
	Total	50	63	37	150

Tabela B.12: Maximização de Expectativas (EM) Weka - BD: Íris com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	50	0	0	50
	2	0	50	0	50
	3	0	14	36	50
	Total	50	64	36	150

B.2 Base de dados Mushroom

Tabela B.13: K-Means - BD: Mushroom original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3424	64	3488
	2	816	1340	2156
	Total	4240	1404	5644

Tabela B.14: K-Means - BD: Mushroom com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3432	56	3488
	2	816	1340	2156
	Total	4248	1396	5644

Tabela B.15: K-Means - BD: Mushroom com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3440	48	3488
	2	744	1412	2156
	Total	4184	1460	5644

Tabela B.16: K-Medoid - BD: Mushroom original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3388	100	3488
	2	810	1346	2156
	Total	4198	1446	5644

Tabela B.17: K-Medoid - BD: Mushroom com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3424	64	3488
	2	816	1340	2156
	Total	4240	1404	5644

Tabela B.18: K-Medoid - BD: Mushroom com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3428	60	3488
	2	816	1340	2156
	Total	4244	1400	5644

Tabela B.19: K-Means Weka - BD: Mushroom original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3432	56	3488
	2	816	1340	2156
	Total	4248	1396	5644

Tabela B.20: K-Means Weka - BD: Mushroom com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3433	55	3488
	2	816	1340	2156
	Total	4249	1395	5644

Tabela B.21: K-Means Weka - BD: Mushroom com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3432	56	3488
	2	816	1340	2156
	Total	4248	1396	5644

Tabela B.22: Maximização de Expectativas (EM) Weka - BD: Mushroom original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3472	16	3488
	2	816	1340	2156
	Total	4288	1356	5644

Tabela B.23: Maximização de Expectativas (EM) Weka - BD: Mushroom com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3472	16	3488
	2	816	1340	2156
	Total	4288	1356	5644

Tabela B.24: Maximização de Expectativas (EM) Weka - BD: Mushroom com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	3472	16	3488
	2	816	1340	2156
	Total	4288	1356	5644

B.3 Base de dados Sonar

Tabela B.25: K-Means - BD: Sonar original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	58	53	111
	2	38	59	97
	Total	96	112	208

Tabela B.26: K-Means - BD: Sonar com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	59	52	111
	2	39	58	97
	Total	98	110	208

Tabela B.27: K-Means - BD: Sonar com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	45	66	111
	2	16	81	97
	Total	61	147	208

Tabela B.28: K-Medoid - BD: Sonar original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	46	65	111
	2	34	63	97
	Total	80	128	208

Tabela B.29: K-Medoid - BD: Sonar com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	54	57	111
	2	37	60	97
	Total	91	117	208

Tabela B.30: K-Medoid - BD: Sonar com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	54	57	111
	2	37	60	97
	Total	91	117	208

Tabela B.31: K-Means Weka - BD: Sonar original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	58	53	111
	2	38	59	97
	Total	96	112	208

Tabela B.32: K-Means Weka - BD: Sonar com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	58	53	111
	2	38	59	97
	Total	96	112	208

Tabela B.33: K-Means Weka - BD: Sonar com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	58	53	111
	2	38	59	97
	Total	96	112	208

Tabela B.34: Maximização de Expectativas (EM) Weka - BD: Sonar original

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	56	55	111
	2	41	56	97
	Total	97	111	208

Tabela B.35: Maximização de Expectativas (EM) Weka - BD: Sonar com normalização linear

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	56	55	111
	2	41	56	97
	Total	97	111	208

Tabela B.36: Maximização de Expectativas (EM) Weka - BD: Sonar com escores-z

Matriz de Confusão Absoluta				
		Predito		
		1	2	Total
Esperado	1	56	55	111
	2	41	56	97
	Total	97	111	208

B.4 Base de dados Vowel

Tabela B.37: K-Means - BD: Vowel original

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	51	18	21	0	0	0	0	0	0	0	0	90
	2	41	24	25	0	0	0	0	0	0	0	0	90
	3	9	31	29	21	0	0	0	0	0	0	0	90
	4	0	0	0	60	0	18	0	0	0	0	12	90
	5	0	0	0	0	46	26	6	0	0	0	12	90
	6	0	0	0	26	15	30	3	0	0	0	16	90
	7	0	11	0	0	23	6	37	0	5	0	8	90
	8	0	6	0	0	0	0	20	16	48	0	0	90
	9	0	2	0	0	0	6	19	8	14	18	23	90
	10	6	12	0	0	0	0	0	12	24	24	12	90
	11	0	0	0	22	0	30	0	0	0	6	32	90
	Total	107	104	75	129	84	116	85	36	91	48	115	990

Tabela B.38: K-Means - BD: Vowel com normalização linear

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	44	28	18	0	0	0	0	0	0	0	0	90
	2	29	34	24	2	0	0	0	0	0	0	1	90
	3	7	14	36	33	0	0	0	0	0	0	0	90
	4	0	0	0	53	0	25	0	0	0	0	12	90
	5	0	0	0	3	41	29	6	4	0	0	7	90
	6	0	0	0	19	32	31	0	0	0	0	8	90
	7	0	0	0	0	23	8	18	25	10	6	0	90
	8	0	0	0	0	0	0	7	34	2	47	0	90
	9	0	0	6	0	0	0	13	18	24	14	15	90
	10	9	0	15	0	0	0	0	9	22	29	6	90
	11	0	0	6	18	14	21	2	6	4	0	19	90
	Total	89	76	105	128	110	114	46	96	62	96	68	990

Tabela B.39: K-Means - BD: Vowel com escores-z

Matriz de Confusão Absoluta													
		Predito											
		1	2	3	4	5	6	7	8	9	10	11	Total
Esperado	1	37	29	18	0	0	0	0	0	0	0	6	90
	2	21	33	25	4	0	0	0	0	0	0	7	90
	3	7	13	35	34	0	1	0	0	0	0	0	90
	4	0	0	0	52	0	26	0	0	0	0	12	90
	5	0	0	0	5	35	24	13	6	0	0	7	90
	6	0	0	0	31	12	29	13	0	0	0	5	90
	7	0	0	0	0	23	6	20	20	21	0	0	90
	8	0	0	0	0	6	0	12	42	30	0	0	90
	9	0	0	2	0	4	5	15	12	16	18	18	90
	10	8	0	16	0	0	0	6	23	7	24	6	90
	11	0	0	1	19	3	29	10	0	0	5	23	90
	Total	73	75	97	145	83	120	89	103	74	47	84	990

Tabela B.40: K-Medoid - BD: Vowel original

Matriz de Confusão Absoluta													
		Predito											
		1	2	3	4	5	6	7	8	9	10	11	Total
Esperado	1	34	35	7	14	0	0	0	0	0	0	0	90
	2	20	16	29	17	0	0	0	0	7	0	1	90
	3	13	0	48	9	0	0	1	0	0	0	19	90
	4	0	0	15	0	0	51	0	0	7	0	17	90
	5	0	0	0	0	27	51	8	1	3	0	0	90
	6	0	0	0	0	8	58	0	0	8	0	16	90
	7	0	0	0	0	22	20	18	25	0	5	0	90
	8	0	0	0	0	1	0	12	77	0	0	0	90
	9	0	0	0	0	0	1	11	35	17	20	6	90
	10	0	6	0	13	0	1	0	35	9	26	0	90
	11	0	0	6	0	0	19	0	0	36	3	26	90
	Total	67	57	105	53	58	201	50	173	87	54	85	990

Tabela B.41: K-Medoid - BD: Vowel com normalização linear

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	28	8	13	0	36	0	0	0	5	0	0	90
	2	21	23	16	4	25	0	0	0	0	0	1	90
	3	10	18	32	23	1	0	0	0	0	0	6	90
	4	0	0	14	65	0	0	0	0	0	0	11	90
	5	0	0	2	30	0	21	6	27	0	0	4	90
	6	0	0	18	35	0	22	6	8	0	0	1	90
	7	5	0	4	9	0	11	12	27	8	14	0	90
	8	0	0	0	0	0	0	14	29	6	41	0	90
	9	2	0	8	0	0	8	5	8	27	14	18	90
	10	6	0	15	0	6	6	0	2	26	23	6	90
	11	0	0	23	21	0	10	0	6	12	0	18	90
	Total	72	49	145	187	68	78	43	107	84	92	65	990

Tabela B.42: K-Medoid - BD: Vowel com escores-z

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	33	9	7	0	0	0	6	0	6	29	0	90
	2	23	23	4	5	0	0	0	0	1	34	0	90
	3	9	15	25	17	0	0	0	0	0	17	7	90
	4	0	0	18	44	0	0	0	0	0	0	28	90
	5	0	0	0	6	28	20	0	1	0	0	35	90
	6	0	0	12	25	9	9	0	2	0	0	33	90
	7	0	0	0	0	17	19	16	34	2	0	2	90
	8	0	0	0	0	6	27	7	49	1	0	0	90
	9	7	0	7	0	0	9	9	28	24	0	6	90
	10	13	0	0	0	2	0	1	32	23	17	2	90
	11	4	0	26	14	0	0	0	0	10	0	36	90
	Total	89	47	99	111	62	84	39	146	67	97	149	990

Tabela B.43: K-Means Weka - BD: Vowel original

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	35	15	27	0	0	0	0	0	13	0	0	90
	2	30	23	32	0	0	0	0	0	3	0	2	90
	3	1	14	41	33	0	0	0	0	1	0	0	90
	4	0	6	0	53	0	25	0	0	0	0	6	90
	5	0	0	0	0	27	25	32	0	0	0	6	90
	6	0	0	2	19	26	31	9	0	0	0	3	90
	7	0	0	0	0	14	9	35	14	11	7	0	90
	8	0	0	0	0	0	1	20	48	6	15	0	90
	9	0	0	6	0	9	0	11	15	3	28	18	90
	10	6	0	12	0	2	0	0	27	6	31	6	90
	11	0	0	6	19	10	23	8	0	0	5	19	90
	Total	72	58	126	124	88	114	115	104	43	86	60	990

Tabela B.44: K-Means Weka - BD: Vowel com normalização linear

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	32	15	31	0	0	0	0	0	12	0	0	90
	2	30	23	36	0	0	0	0	0	1	0	0	90
	3	2	14	41	32	0	0	0	0	1	0	0	90
	4	0	0	0	53	0	25	0	0	0	0	12	90
	5	0	0	0	2	42	24	9	0	2	0	11	90
	6	0	0	0	19	30	31	0	0	0	0	10	90
	7	0	0	0	0	25	6	35	6	18	0	0	90
	8	0	0	0	0	0	0	37	46	7	0	0	90
	9	0	0	6	0	0	0	20	17	9	22	16	90
	10	6	0	10	0	0	0	6	30	2	30	6	90
	11	0	0	0	9	16	25	0	0	0	21	19	90
	Total	70	52	124	115	113	111	107	99	52	73	74	990

Tabela B.45: K-Means Weka - BD: Vowel com escores-z

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	39	30	15	0	0	0	0	0	6	0	0	90
	2	35	32	6	0	0	0	0	0	0	0	17	90
	3	15	1	12	31	0	26	0	0	1	0	4	90
	4	0	0	0	52	1	25	0	0	0	0	12	90
	5	0	0	0	6	39	3	28	0	9	0	5	90
	6	0	0	0	20	36	24	0	0	0	0	10	90
	7	0	0	0	0	27	0	31	6	18	8	0	90
	8	0	0	0	0	0	0	33	46	9	2	0	90
	9	1	1	0	0	0	5	19	19	12	18	15	90
	10	18	12	0	0	0	0	6	28	0	24	2	90
	11	0	0	0	10	6	27	3	0	0	15	29	90
	Total	108	76	33	119	109	110	120	99	55	67	94	990

Tabela B.46: Maximização de Expectativas (EM) Weka - BD: Vowel original

Matriz de Confusão Absoluta													
		Predito											Total
		1	2	3	4	5	6	7	8	9	10	11	
Esperado	1	35	15	14	0	0	0	0	0	0	0	26	90
	2	22	21	19	0	0	0	0	0	0	0	28	90
	3	6	15	36	31	0	0	0	0	0	0	2	90
	4	0	6	0	44	0	34	0	0	0	0	6	90
	5	0	0	0	0	56	18	7	0	6	0	3	90
	6	0	0	0	15	35	30	4	0	6	0	0	90
	7	0	0	0	0	22	0	44	6	18	0	0	90
	8	0	0	0	0	0	0	36	48	6	0	0	90
	9	1	0	0	0	0	5	25	14	13	18	14	90
	10	11	0	7	0	0	0	15	27	0	18	12	90
	11	0	0	0	24	15	28	0	0	0	0	23	90
	Total	75	57	76	114	128	115	131	95	49	36	114	990

Tabela B.47: Maximização de Expectativas (EM) Weka - BD: Vowel com normalização linear

Matriz de Confusão Absoluta													
		Predito											
		1	2	3	4	5	6	7	8	9	10	11	Total
Esperado	1	35	15	15	0	0	0	0	0	0	0	25	90
	2	21	21	20	0	0	0	0	0	0	0	28	90
	3	6	15	36	32	0	0	0	0	0	0	1	90
	4	0	0	0	59	0	19	0	0	5	0	7	90
	5	0	0	0	4	49	12	10	0	11	0	4	90
	6	0	0	0	34	11	23	6	0	15	0	1	90
	7	0	0	0	0	27	1	38	6	18	0	0	90
	8	0	0	0	0	0	0	36	48	6	0	0	90
	9	0	0	0	0	0	6	25	11	16	18	14	90
	10	0	0	12	0	0	0	15	25	2	24	12	90
	11	0	0	0	30	1	24	0	0	10	0	25	90
	Total	62	51	83	159	88	85	130	90	83	42	117	990

Tabela B.48: Maximização de Expectativas (EM) Weka - BD: Vowel com escores-z

Matriz de Confusão Absoluta													
		Predito											
		1	2	3	4	5	6	7	8	9	10	11	Total
Esperado	1	34	15	15	0	0	0	0	0	0	0	26	90
	2	21	21	20	0	0	0	0	0	0	0	28	90
	3	6	25	36	21	0	0	0	0	0	0	2	90
	4	0	6	0	62	0	16	0	0	0	0	6	90
	5	0	0	0	3	50	12	10	0	11	0	4	90
	6	0	0	0	34	13	23	4	0	15	0	1	90
	7	0	0	0	0	28	1	37	6	18	0	0	90
	8	0	0	0	0	0	0	36	48	6	0	0	90
	9	0	0	0	0	0	6	25	12	15	18	14	90
	10	0	0	12	0	0	0	15	26	1	24	12	90
	11	0	0	0	29	2	24	0	0	10	0	25	90
	Total	61	67	83	149	93	82	127	92	76	42	118	990

B.5 Base de dados Wine

Tabela B.49: K-Means - BD: Wine original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	46	0	13	59
	2	1	50	20	71
	3	0	19	29	48
	Total	47	69	62	178

Tabela B.50: K-Means - BD: Wine com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	2	65	4	71
	3	0	0	48	48
	Total	61	65	52	178

Tabela B.51: K-Means - BD: Wine com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	3	66	2	71
	3	0	0	48	48
	Total	62	66	50	178

Tabela B.52: K-Medoid - BD: Wine original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	48	0	11	59
	2	2	49	20	71
	3	0	16	32	48
	Total	50	65	63	178

Tabela B.53: K-Medoid - BD: Wine com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	10	58	3	71
	3	0	0	48	48
	Total	69	58	51	178

Tabela B.54: K-Medoid - BD: Wine com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	17	53	1	71
	3	0	0	48	48
	Total	76	53	49	178

Tabela B.55: K-Means Weka - BD: Wine original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	6	62	3	71
	3	0	0	48	48
	Total	65	62	51	178

Tabela B.56: K-Means Weka - BD: Wine com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	2	63	6	71
	3	0	0	48	48
	Total	61	63	54	178

Tabela B.57: K-Means Weka - BD: Wine com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	59	0	0	59
	2	2	63	6	71
	3	0	0	48	48
	Total	61	63	54	178

Tabela B.58: Maximização de Expectativas (EM) Weka - BD: Wine original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	57	2	0	59
	2	0	68	3	71
	3	0	0	48	48
	Total	57	70	51	178

Tabela B.59: Maximização de Expectativas (EM) Weka - BD: Wine com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	56	3	0	59
	2	0	68	3	71
	3	0	0	48	48
	Total	56	71	51	178

Tabela B.60: Maximização de Expectativas (EM) Weka - BD: Wine com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	57	2	0	59
	2	0	68	3	71
	3	0	0	48	48
	Total	57	70	51	178

B.6 Base de dados WNBA

Tabela B.61: K-Means - BD: WNBA original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	43	9	1	53
	2	8	35	4	47
	3	0	13	7	20
	Total	51	57	12	120

Tabela B.62: K-Means - BD: WNBA com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	44	9	0	53
	2	6	36	5	47
	3	0	8	12	20
	Total	50	53	17	120

Tabela B.63: K-Means - BD: WNBA com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	8	0	53
	2	6	37	4	47
	3	0	9	11	20
	Total	51	54	15	120

Tabela B.64: K-Medoid - BD: WNBA original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	7	1	53
	2	8	35	4	47
	3	0	13	7	20
	Total	53	55	12	120

Tabela B.65: K-Medoid - BD: WNBA com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	48	5	0	53
	2	7	34	6	47
	3	0	8	12	20
	Total	55	47	18	120

Tabela B.66: K-Medoid - BD: WNBA com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	48	5	0	53
	2	7	34	6	47
	3	0	8	12	20
	Total	55	47	18	120

Tabela B.67: K-Means Weka - BD: WNBA original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	44	9	0	53
	2	6	36	5	47
	3	0	8	12	20
	Total	50	53	17	120

Tabela B.68: K-Means Weka - BD: WNBA com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	8	0	53
	2	6	35	6	47
	3	0	8	12	20
	Total	51	51	18	120

Tabela B.69: K-Means Weka - BD: WNBA com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	8	0	53
	2	6	37	4	47
	3	0	10	10	20
	Total	51	55	14	120

Tabela B.70: Maximização de Expectativas (EM) Weka - BD: WNBA original

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	7	1	53
	2	6	37	4	47
	3	0	11	9	20
	Total	51	55	14	120

Tabela B.71: Maximização de Expectativas (EM) Weka - BD: WNBA com normalização linear

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	7	1	53
	2	6	37	4	47
	3	0	11	9	20
	Total	51	55	14	120

Tabela B.72: Maximização de Expectativas (EM) Weka - BD: WNBA com escores-z

Matriz de Confusão Absoluta					
		Predito			
		1	2	3	Total
Esperado	1	45	7	1	53
	2	6	37	4	47
	3	0	11	9	20
	Total	51	55	14	120