UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

ÉVERTON MARANGONI GAVA

O MÉTODO DE DENSIDADE PELO ALGORITMO *DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE* (DBSCAN) NA TAREFA DE CLUSTERIZAÇÃO DA *SHELL ORION DATA MINING ENGINE*

CRICIÚMA, DEZEMBRO DE 2011

ÉVERTON MARANGONI GAVA

O MÉTODO DE DENSIDADE PELO ALGORITMO *DENSITY-BASED SPATIAL CLUSTERING OF APPLICATIONS WITH NOISE* (DBSCAN) NA TAREFA DE CLUSTERIZAÇÃO DA *SHELL ORION DATA MINING ENGINE*

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientadora: Profª. MSc. Merisandra Côrtes de Mattos.

CRICIÚMA, DEZEMBRO DE 2011

ÉVERTON MARANGONI GAVA

O Método de Densidade pelo Algoritmo *Density-Based Spatial Clustering Of Applications With Noise* (DBSCAN) na Tarefa De Clusterização da *Shell Orion Data Mining Engine*

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Profa. MSc. Merisandra Cortês de Mattos (UNESC)
Orientadora

Prof. MSc. Kristian Madeira (UNESC)

Prof. Esp. Fábio Bif Goulart (UNESC)

RESUMO

A grande quantidade de dados que é gerada e armazenada nas mais diversas áreas de conhecimento, torna necessário o desenvolvimento de tecnologias destinadas à análise de informações, possibilitando a obtenção de novos conhecimentos. Dentre essas tecnologias, destaca-se o *data mining*, que por meio da aplicação de algoritmos com finalidades específicas, tenta extrair um conjunto de padrões possivelmente existentes no conjunto de dados, sendo que para isso são utilizadas ferramentas computacionais que em sua maioria são proprietárias. Considerando isso, o Grupo de Pesquisa em Inteligência Computacional Aplicada do Curso de Ciência da Computação da UNESC, mantém em desenvolvimento o projeto da *Shell Orion Data Mining Engine* que implementa diversos métodos e tarefas de *data mining*. Objetivando ampliar as funcionalidades da *Shell Orion*, essa pesquisa consiste na implementação e na demonstração de funcionamento do algoritmo *Density-Based Spatial Clustering of Applications With Noise* (DBSCAN) que utiliza o conceito de *cluster* baseado em densidade para a tarefa de clusterização, que tem como objetivo particionar um conjunto de dados em grupos distintos. Considerando a utilização do método de densidade, o algoritmo DBSCAN realiza a clusterização procurando por regiões densas no espaço dos dados, permitindo que sejam encontrados grupos com formatos arbitrários e sejam detectados *outliers*. Ao final da pesquisa, diversos testes foram efetuados, e o desempenho do algoritmo desenvolvido foi avaliado por meio de medidas estatísticas que comprovaram o correto funcionamento do DBSCAN na *Shell Orion Data Mining Engine*.

Palavras – Chave: Inteligência Computacional, *Data Mining*, Clusterização, Método de Densidade, Algoritmo DBSCAN, Detecção de *Outliers*.

ABSTRACT

The large quantity of data which is produced and stocked in several areas of knowledge needs a development of the technologies, related to analysis of the informations, showing a way to bring new knowledge. Among these technologies, data mining points out that through practicing algorithm with special purposes, tries to draw out a standard set possibily being in data set and some computation tools are put to use where most of them are owner. Taking into consideration this, the Research Group at Applicated Computation Intelligence Course of Computation Science at UNESC, keeps in force the development of Shell Orion Data Mining Engine project which establishes many methods and tasks of data mining. There is a purpose to increase the operations of Shell Orion, this research consists of establishing and an exhibition of algorithm operation Density - Based Spatial Clustering of Applications with Noise (DBSCAN) which makes use of the concept clustering based on density for the clustering operation whose purpose is to take part in data sets from different sets. According to the method of density, DBSCAN Algorithm takes effect the cluster data looking for heavy areas in the space of datas, tolerating which may be found series of arbitrary shapes, and outliers are detected. At the end of the research, several tests were done, and the practice of algorithm developed was rated through statistical measures which confirmed the correct activity of DBSCAN on Shell Orion Data Mining Engine.

Keywords: Computation Intelligence, Data Mining, Clustering, Density Method, Algorithm DBSCAN, Outlier Detection.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

AGNES	*Agglomerative Nesting*
API	*Application Programming Interface*
BBS	*Biasis Box Sampling*
DBSCAN	*Density-Based Spatial Clustering of Applications with Noise*
DENCLUE	*Density-Based Clustering*
DIANA	*Divisive Analysis*
DM	*Data Mining*
FCM	*Fuzzy C-Means*
GK	*Gustafson-Kessel*
GTA	Grupo Técnico de Assessoramento
JDBC	*Java Database Connectivity*
KDD	*Knowledge Discovery in Databases*
OPTICS	*Ordering Points to Identify the Clustering Structure*
PC	Componentes Principais
PCA	Principal Component Analysis
RCP	*Robust C-Prototypes*
RNA	Redes Neurais Artificiais
SGBD	Sistemas Gerenciadores de Banco de Dados
SOM	*Self-Organizing Maps*
SQL	*Structured Query Language*
UNESC	Universidade do Extremo Sul Catarinense
URCP	*Unsupervised Robust C-Prototypes*

SUMÁRIO

1 INTRODUÇÃO

Progressos constantes no processo de aquisição e armazenamento digital de informações permitiram que as mais variadas instituições formassem grandes bases de dados. A fim de facilitar o processo de descoberta de conhecimento, técnicas capazes de extrair informações significativas destes vastos repositórios foram desenvolvidas, sendo que a procura por relações úteis entre os dados ficou conhecida como *Knowledge Discovery in Databases* (KDD), sendo o *data mining* a principal etapa desse processo.

O *data mining* é uma área interdisciplinar que reúne técnicas de aprendizado de máquina, reconhecimento de padrões, estatística, bancos de dados, visualização de dados, entre outras, para abordar a questão da descoberta de conhecimento em bases de dados (HAN; KAMBER, 2006, tradução nossa; SIVANANDAM; SUMATHI, 2006, tradução nossa).

Existem diversas tarefas de *data mining*, sendo que a escolha de uma determinada tarefa depende do resultado que se deseja obter e do tipo do conjunto de dados que se tem a disposição (FAYYAD; PIATETSKY-SHAPIRO; SMYTH ,1996, tradução nossa).

Portanto, para execução do *data mining***,** além das tarefas, são necessários métodos que as implementem, sendo estes compostos por diferentes algoritmos, disponibilizados em ferramentas computacionais específicas**.** Contudo, em sua grande maioria essas ferramentas são proprietárias.

Considerando isso, o Grupo de Pesquisa em Inteligência Computacional Aplicada do Curso de Ciência da Computação da UNESC mantém em desenvolvimento o projeto da *Shell Orion Data Mining Engine*, onde são disponibilizados diversas tarefas de *data mining*. A *Shell Orion* possui implementada as tarefas de classificação (algoritmos ID3,

C4.5, CART e RBF) , de associação (algoritmo Apriori) e de clusterização (algoritmos *K-Means*, *Kohonen*, *Fuzzy C-Means*, *Gustafson-Kessel*, *Gath-Geva*, *Robust C-Prototypes* e *Unsupervised Robust C-Prototypes*).

A tarefa de clusterização tem por objetivo agrupar objetos similares, de acordo com suas características, em subconjuntos ou *clusters* relativamente homogêneos. Um *cluster* consiste em uma coleção de objetos que são similares entre si e dissimilares entre objetos de outros *clusters* (LAROSE, 2005, tradução nossa).

Existem diversos métodos de clusterização, dentre os quais podem ser destacados os métodos hierárquicos, de particionamento, métodos baseados em grade, em modelos e os métodos baseados em densidade (HAN; KAMBER, 2006, tradução nossa; JAIN; MURTY; FLYNN, 1999, tradução nossa).

Métodos tradicionais de clusterização, como os de particionamento, geralmente enfrentam dificuldades para encontrar agrupamentos com formatos arbitrários e não retornam bons resultados quando a base de dados em questão está contaminada com *outliers* (dados que destoam do padrão geral da distribuição). Outro ponto a ser considerado em alguns destes métodos, é que o usuário tem a necessidade de informar previamente o número de *clusters* que serão gerados, o que na maioria das vezes não é uma tarefa simples (ESTER et al, 1996, tradução nossa; HAN; KAMBER, 2006, tradução nossa).

A fim de minimizar esses problemas tem-se os métodos baseados em densidade que são usados para a detecção de agrupamentos com formatos arbitrários em conjuntos de dados contendo *outliers*. Para algoritmos que adotam essa abordagem, *clusters* são regiões com alta densidade de pontos no espaço dos dados, separadas de outras regiões densas, por regiões de baixa densidade (que representam *outliers*). Por sua vez, essas regiões de alta densidade podem conter formato arbitrário no espaço de dados (ANKERST et al,1999, tradução nossa; KRIEGEL et al, 2011, tradução nossa).

Algoritmos que encontram *clusters* baseados em densidade não necessitam que seja informado de maneira prévia o número de grupos a serem formados, e não fazem suposições sobre a variância ou a distribuição interna dos objetos nos possíveis grupos que possam vir a existir no conjunto de dados. Essas propriedades permitem que sejam encontrados agrupamentos baseados nas propriedades dos dados, o que consequentemente impõe uma estruturação menos rígida aos objetos (KRIEGEL et al, 2011, tradução nossa).

Utilizando esta abordagem, esta pesquisa consiste no desenvolvimento do algoritmo *Density-Based Spatial Clustering of Applications With Noise* (DBSCAN) para a tarefa de clusterização *na Shell Orion Data Mining Engine.*

1.1 OBJETIVO GERAL

Desenvolver o método baseado em densidade, por meio do algoritmo DBSCAN, na tarefa de clusterização da *Shell Orion Data Mining Engine*.

1.2 OBJETIVOS ESPECÍFICOS

Entre os objetivos específicos desta pesquisa estão:

a) compreender os principais conceitos de *data mining* e a tarefa de clusterização;
b) entender o método baseado em densidade e o algoritmo DBSCAN;
c) aplicar o algoritmo DBSCAN na tarefa de clusterização de dados da *Shell Orion Data Mining Engine*;
d) demonstrar matematicamente o funcionamento do algoritmo DBSCAN;

e) analisar o desempenho do algoritmo desenvolvido por meio de medidas estatísticas.

1.3 JUSTIFICATIVA

O *data mining* é o processo na etapa de descoberta de conhecimento em bases de dados que consiste na aplicação de algoritmos específicos, que sob limitações de eficiência computacional aceitáveis, tanto de tempo quanto de processamento, produzem uma enumeração particular de padrões sobre os dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, tradução nossa).

Os padrões encontrados pelo processo de *data mining* podem auxiliar na previsão de um conhecimento futuro e ser de fundamental importância na tomada de decisões estratégicas.

A fim de auxiliar as instituições no processo de descoberta de conhecimento, são usadas ferramentas denominadas *Shells*, porém, existe certa carência destas ferramentas que sejam gratuitas. O projeto da *Shell Orion Data Mining Engine* implementa as tarefas consideradas mais importantes no processo de *data mining*, por meio de vários métodos, em uma ferramenta gratuita. Esta pesquisa amplia as funcionalidades da *Shell Orion*, acrescentando o método baseado em densidade na tarefa de clusterização, por meio do algoritmo DBSCAN.

A clusterização pode ser vista como uma das tarefas básicas no processo de *data mining*, permitindo identificar os grupos existentes em um conjunto de objetos, assim auxiliando na estruturação e na compreensão do conjunto de dados original. Os resultados da tarefa de clusterização também podem ser utilizados por outras técnicas de *data mining*, tais

como a classificação e a sumarização, que realizariam seu trabalho nos *clusters* encontrados (CARLANTONIO, 2001; HAN; KAMBER, 2006, tradução nossa).

Dentre os fatores que motivaram a escolha do método de densidade para a tarefa de clusterização nesta pesquisa, podem ser destacados a possibilidade de encontrar *clusters* com formatos diversos, o fato de não ser necessário informar com antecedência o número de agrupamentos que serão gerados e a capacidade de encontrar *outliers* no conjunto de dados. Outro fator considerado foi que conjuntos de dados reais usualmente apresentam agrupamentos com densidades e formas distintas, além de possuírem quantidade significativa de elementos considerados *outliers*, o que torna desejável a utilização de métodos eficientes para lidar com esse tipo de cenário (APPEL, 2010; ESTER et al, 1996, tradução nossa; GAN; MA; WU, 2007, tradução nossa; KRIEGEL et al, 2011, tradução nossa).

O algoritmo DBSCAN encontra agrupamentos baseado na vizinhança dos objetos, onde a densidade associada a um ponto é obtida por meio da contagem do número de pontos vizinhos em uma determinada região ao redor desse ponto (ERTÖZ; STEINBACH; KUMAR, 2006, tradução nossa). Esse algoritmo possui a capacidade de encontrar *clusters* considerando as propriedades dos dados, pois não requer que seja informado antecipadamente o número de *clusters*, permitindo a formação de grupos com formatos arbitrários. Outras características importantes do algoritmo são a capacidade de identificar *outliers* e a possibilidade de poder trabalhar com diversas medidas de distância (ANKERST et al, 1999, tradução nossa; ESTER et al, 1996, tradução nossa; METZ, 2006).

1.4 ESTRUTURA DO TRABALHO

Esta pesquisa é composta por seis capítulos, sendo no Capítulo 1 contextualizado

o tema proposto, os objetivos pretendidos e a justificativa para essa pesquisa.

No Capítulo 2 estão expostos os principais conceitos relacionados ao processo de KDD, bem como os ligados à etapa de *data mining*. A *Shell Orion* também é abordada nessa parte do trabalho.

A tarefa de clusterização em *data mining* é o tema do Capítulo 3. O método de clusterização baseada em densidade e alguns algoritmos que o implementam também são descritos nesse capítulo.

O algoritmo DBSCAN, bem como seu funcionamento, é definido no Capítulo 4. Ainda é demonstrada nessa parte, uma heurística que auxilia na escolha dos parâmetros de entrada do algoritmo. O Capítulo 5 apresenta alguns trabalhos correlatos que usaram o algoritmo DBSCAN.

No Capítulo 6 são descritas as etapas do trabalho desenvolvido, a metodologia utilizada e os resultados obtidos pelo módulo do algoritmo DBSCAN.

Finalizando, tem-se a conclusão da pesquisa e sugestões de trabalhos futuros.

2 DESCOBERTA DE CONHECIMENTO EM BASES DE DADOS

O progresso nas tecnologias de armazenamento e aquisição de dados digitais resultou em crescimento das bases de dados. Embora sabendo que padrões interessantes e informações potencialmente úteis podem ser extraídas desses repositórios, o volume de dados torna difícil, senão impraticável, a busca por esse conhecimento implícito sem o auxílio de ferramentas computacionais (ZHOU, 2003, tradução nossa).

Nesse contexto complexo em que existe uma sobrecarga considerável de dados nos mais variados ramos de conhecimento da sociedade, surge uma nova área de pesquisa denominada Descoberta de Conhecimento em Bases de Dados (*Knowledge Discovery in Databases* - KDD), que envolve a aplicação de tecnologias computacionais para resolver o problema de extração de conhecimento em bases de dados (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, tradução nossa).

Existem diversas definições distintas para o processo de KDD na literatura. Fayyad, Piatetsky-Shapiro e Smyth (1996, tradução nossa) definem o KDD como o processo não trivial de identificação de padrões válidos, novos, potencialmente úteis e finalmente compreensíveis a partir de um conjunto de dados. Já Frietman, Hill e Khoe (2001, tradução nossa) consideram o KDD como um processo automático de análise exploratória de grandes bases de dados. Para Cabena et al (1998, tradução nossa) o processo de KDD significa extrair, de grandes conjuntos de dados, sem nenhuma formulação de hipóteses previamente definidas, informações relevantes e novas que podem ser usadas no processo de tomada de decisão. Seja como for, todas as definições concordam que o objetivo final do processo é trazer à tona novos conhecimentos, que possam vir a ser úteis, a partir de um domínio de aplicação específico.

Entre as características do processo de KDD estão ser um processo interativo e

iterativo, composto por várias etapas. O termo interativo indica a necessidade de atuação do homem como responsável por controlar o processo, pois o mesmo envolve um número elevado de decisões a serem tomadas (SASSI, 2006). É um processo iterativo, por que durante a sua realização, pode existir a possibilidade de repetições integrais ou parciais na busca de resultados satisfatórios por meio de sucessivos refinamentos (GOLDSCHIMIDT; PASSOS, 2005).

Considerando a natureza interdisciplinar do processo de KDD, têm-se várias etapas aplicadas sucessivamente para se chegar ao resultado esperado, ou seja, a extração do conhecimento implícito em bases de dados. Cada etapa do processo possui uma intersecção com as demais, desse modo, os resultados obtidos em uma fase são utilizados para melhorar os resultados da próxima (SASSI, 2006):

a) **seleção dos dados:** a etapa de seleção dos dados, também conhecida por redução de dados, consiste na criação de um conjunto de dados-alvo ou dados selecionados. Nesta etapa do processo, seleciona-se um conjunto de dados ou um conjunto de atributos desses dados que serão fornecidos para os algoritmos de *data mining* (DM). Em essência, consiste na identificação de quais informações, dentre as existentes, devem ser efetivamente consideradas durante o processo de KDD (GOLDSCHIMIDT; PASSOS, 2005). Uma das motivações para essa etapa é que ela otimiza o tempo de processamento dos algoritmos de DM, visto que eles atuam em um subconjunto representativo da base de dados em questão, diminuindo seu espaço de busca (SASSI,2006);
b) **pré-processamento dos dados:** busca-se aprimorar a qualidade dos dados coletados na etapa de seleção de dados, a fim de assegurar a qualidade dos fatos por eles representados. Frequentemente os dados apresentam vários problemas, tais como a grande quantidade de valores desconhecidos, *outliers*,

entre outros problemas (BATISTA, 2003). Essa etapa envolve a verificação da consistência dos dados, a correção de possíveis erros, a eliminação de registros duplicados e o preenchimento ou a eliminação de valores nulos ou redundantes (SASSI, 2006);

c) **transformação dos dados:** também denominada de codificação de dados, visa principalmente converter o conjunto de dados brutos selecionados na etapa de pré–processamento, em uma forma padrão de uso. Pode ser necessário transformar a forma em que os dados estão representados, objetivando superar limitações existentes em métodos na etapa subseqüente de DM. Entre as vantagens de se codificar um atributo estão: melhorar a compreensão do conhecimento gerado, diminuir o tempo de processamento da técnica de DM usada, entre outras (PYLE, 1999, tradução nossa; SASSI, 2006);

d) ***data mining:*** é a principal etapa do KDD. Nesta etapa ocorre a busca efetiva por novos padrões que possam gerar conhecimento útil a partir dos dados (GOLDSCHIMIDT; PASSOS, 2005). É caracterizada pela existência do algoritmo minerador, que diante da tarefa especificada será capaz de extrair conhecimento útil e implícito em conjuntos de dados selecionados, pré-processados e transformados;

e) **interpretação e avaliação do conhecimento:** usualmente denominada como pós-processamento, envolve a visualização, análise e a interpretação do conhecimento gerado na etapa de DM (GOLDSCHIMIDT; PASSOS, 2005). Geralmente a principal meta dessa etapa consiste em melhorar a compreensão do conhecimento gerado, validando-o pela concepção de um analista de dados e por medidas de qualidade (SASSI, 2006). Considerando

> que em KDD o resultado deve ser compreensível ao usuário, podem se recorrer à utilização de técnicas de visualização de dados para a finalidade, visto que essas técnicas estimulam a percepção e a inteligência humana, aumentando a capacidade de entendimento e a associação de novos padrões (BIGUS, 1996, tradução nossa; GOLDSCHIMIDT; PASSOS, 2005).

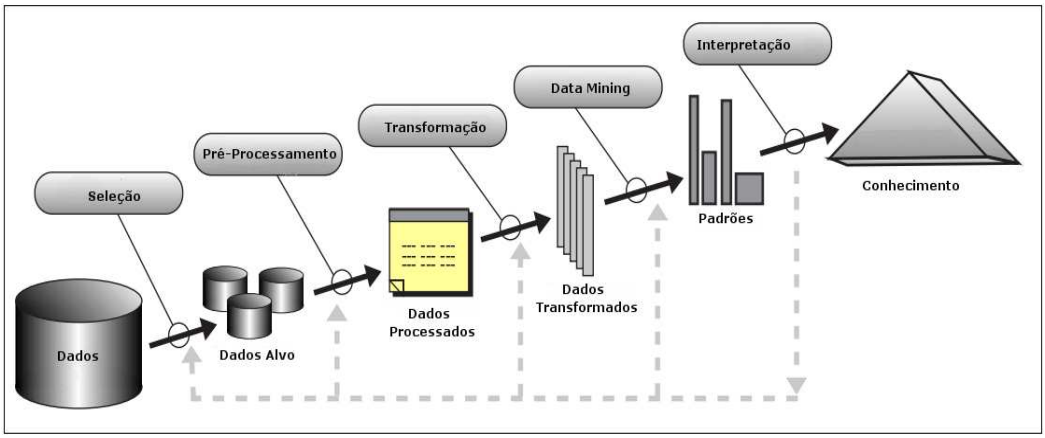

Figura 1. Etapas do processo de descoberta de conhecimento
Fonte: Adaptado de FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. (1996)

Dentre todas as etapas, a mais importante no âmbito desse trabalho é a de DM. Enquanto as etapas de seleção, pré-processamento e transformação estão mais diretamente ligadas à preparação, tratamento de imperfeições e qualidade dos dados, a etapa de DM efetivamente realizará a busca por padrões potencialmente interessantes no conjunto de dados selecionado (DASU; JOHNSON, 2003, tradução nossa; PYLE, 1999, tradução nossa).

2.1 *DATA MINING*

Muitos autores referem-se a DM e ao processo de KDD de forma indistinta. O termo DM faz referência à etapa em que são aplicados algoritmos específicos para efetivamente extrair padrões dos dados, enquanto o termo KDD faz referência ao processo de descoberta de padrões úteis nos dados em geral, englobando etapas adicionais, que são essenciais para a interpretação adequada e avaliação da qualidade do conhecimento obtido na

etapa de DM (FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996, tradução nossa).

O DM pode ser definido como o processo de explorar e analisar grandes conjuntos de dados, extraindo informação e conhecimento sob a forma de novas relações e padrões úteis na resolução de problemas de um domínio de aplicação específico (SIVANANDAM; SUMATTI, 2006, tradução nossa).

O DM pode ser aplicado em diversos campos de pesquisa distintos, dentre os quais se podem destacar (WITTEN; FRANK; HALL, 2011, tradução nossa):

a) **mineração de dados na web:** os motores de busca da internet como *Google, Yahoo, Ask, Bing* entre outros, fazem uso de técnicas de DM nos conteúdos pesquisados pelos usuários de seus serviços, e com base nisso selecionam anúncios que cada usuário individual possa estar interessado, aumentando assim, as chances de um determinado usuário acessar um determinado serviço. Esses provedores de busca possuem apelo para aprimorar suas técnicas na área, pois os investidores, ou anunciantes de suas páginas, os pagam apenas se os usuários clicam em seus *links*;
b) **segmentação de imagens de satélite:** técnicas de DM podem ser utilizadas para detectar manchas de óleo provenientes de imagens de satélite, a fim de prever antecipadamente desastres ecológicos e coibir derramamentos ilegais. Dada a grande quantidade de imagens geradas e a dificuldade de classificação manual dessas áreas, as técnicas de DM funcionam como um filtro para os usuários finais desses sistemas, reduzindo o espectro de busca e o número de alarmes falsos, o que dependendo do caso, pode gerar economia;
c) **previsão de carga do sistema elétrico:** no setor energético, é de suma importância a previsão de demanda futura com a maior antecedência

possível. Quanto mais precisas forem as estimativas de carga máxima e mínima para períodos de curto e longo prazo, mais significativas serão as economias para as companhias geradoras e distribuidoras, visto que essa antecipação de previsão futura gera um melhor planejamento. Técnicas de DM podem ser utilizadas para gerar sistemas de previsão de carga, com detalhes de horas, alimentados com dados estatísticos históricos. Acompanhados por especialistas humanos, esses sistemas podem auxiliar decisivamente nas tomadas de decisões dessas companhias;

d) ***marketing* e vendas:** esse é um domínio de aplicação tradicional do DM**,** pois nessas áreas tem-se grandes bases de dados, até pouco tempo intocadas, as quais constituem-se em valiosos ativos. As aplicações voltadas ao *marketing* e vendas objetivam gerar previsões futuras. O DM pode determinar grupos para os quais novos serviços podem ser direcionados, tais como conjuntos de clientes rentáveis, clientes fiéis, clientes que preferem utilizar dinheiro em espécie ao invés de cartão de crédito, entre várias outras funcionalidades.

Considerando a vastidão de domínios de aplicação do DM, e que os usuários do conhecimento gerado pelo processo de KDD podem estar interessados em tipos distintos de padrões sobre os dados, existem diversas tarefas de DM, sendo que a escolha de uma determinada tarefa depende do conhecimento que se deseja obter.

2.1.1 Tarefas e Métodos de *Data Mining*

As tarefas de DM são usadas conforme o padrão dos dados que se deseja obter, e em geral, no seu mais alto nível, podem ser classificadas em duas categorias (Figura 2):

descritivas e preditivas. As tarefas preditivas buscam informar o valor de um atributo com base em outros atributos existentes. Já as tarefas descritivas procuram caracterizar as propriedades gerais dos dados, baseando-se nas semelhanças, ou diferenças de padrões existentes entre esses dados (HAN; KAMBER, 2006, tradução nossa).

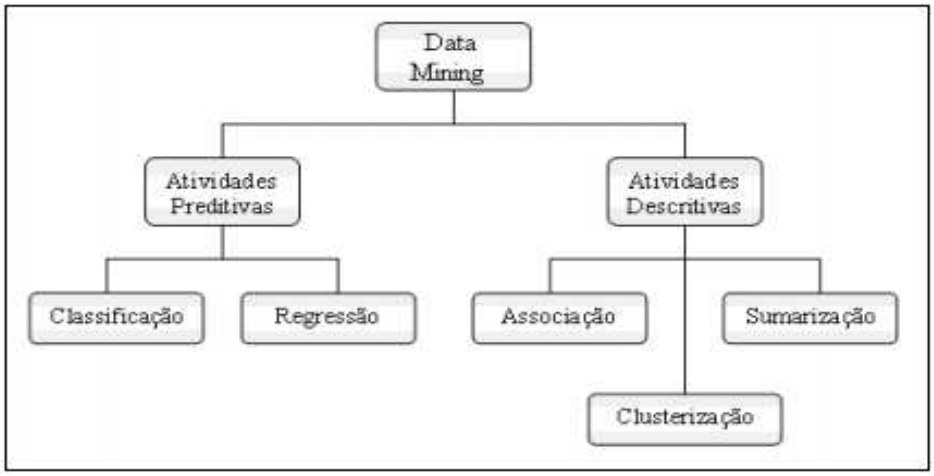

Figura 2. Tarefas de *Data Mining*
Fonte: Scotti, A. (2010)

As principais tarefas de DM, seus respectivos objetivos bem com alguns exemplos práticos de sua aplicação são descritos a seguir:

a) **classificação:** de acordo com Goldschimidt e Passos (2005) a classificação é uma das tarefas mais comuns e importantes em DM. Essa tarefa preditiva consiste em construir uma função que possa ser aplicada a dados não classificados visando categoralizá-los em classes pré-definidas (HARRISON, 1998, tradução nossa). Pode ser usada para identificar transações fraudulentas de cartão de crédito, classificar ações da bolsa de valores em grupos de lucros potenciais baixos, médios ou altos, entre inúmeras aplicações (FAWCETT; PROVOST, 1997, tradução nossa; OLIVEIRA, 2008);

b) **regressão:** essa tarefa preditiva é similar a classificação, sendo que o diferencial entre as duas é que a regressão trabalha apenas com atributos numéricos (GOLDSCHMIDT; PASSOS, 2005). A regressão lida com

resultados contínuos, enquanto que a classificação lida com resultados discretos. Esta tarefa tem como objetivo determinar o valor mais provável de algum índice diante de dados do passado ou de outros índices semelhantes sobre os quais se tem conhecimento (OLIVEIRA, 2008). Pode ser usada para estimativa do tempo de vida de uma pessoa, do número de filhos em uma família, de demanda de um novo produto, entre outras aplicações distintas;

c) **associação:** tarefa que tem por objetivo descrever as relações de associação entre diferentes itens de uma transação na base de dados (PAL; MITRA, 2004, tradução nossa). A associação busca por itens que tendem a ocorrer juntos em uma mesma transação, assim caracterizando padrões e tendências. Redes de varejo podem usar esta tarefa para planejar a disposição dos produtos nas prateleiras das lojas ou em um catálogo, de modo que os itens geralmente adquiridos na mesma compra sejam vistos próximos entre si (HARRISON, 1998, tradução nossa);

d) **sumarização:** conforme Fayyad, Piatetsky-Shapiro e Smyth (1996, tradução nossa) a sumarização envolve métodos para encontrar uma descrição compacta para um subconjunto de dados. Exemplos de técnicas de sumarização incluem medidas de posição, variabilidade, histogramas, *boxplots* e diagramas de dispersão (SFERRA; CORRÊA, 2003). Editoras podem utilizar a sumarização para buscar por características comuns a boa parte dos clientes, como por exemplo, são assinantes da revista X, mulheres entre 20 e 30 anos, com nível superior e que trabalham na área de finanças. Tal informação poderia ser usada pelo departamento de *marketing* para direcionar ofertas a novos assinantes (GOLDSCHIMDT; PASSOS, 2005);

e) **clusterização:** consiste na separação de um conjunto de dados em grupos, de

> modo que objetos dentro de um grupo sejam altamente similares entre si e possuem dissimilaridade com objetos de outros grupos (HAN; KAMBER, 2006, tradução nossa). Diferentemente da tarefa de classificação, que possui classes pré-definidas, a clusterização precisa identificar automaticamente essas classes, utilizando alguma medida de similaridade (GOLDSCHIMIDT; PASSOS, 2005). Como exemplo de uso dessa tarefa pode-se citar a identificação de áreas do solo que apresentam uso similar em bases de dados geográficas (HAN; KAMBER, 2006, tradução nossa).

Como são diversas as tarefas possíveis em DM, então naturalmente existem muitos métodos disponíveis para auxiliar na implementação dessas tarefas, sendo que a escolha do método mais adequado depende das necessidades e dos resultados desejados.

Alguns dos métodos comumente utilizados são:

a) **redes neurais artificiais (RNA):** são técnicas computacionais inspiradas no sistema nervoso biológico, cujo funcionamento é semelhante a alguns procedimentos humanos, ou seja, aprendem pela experiência, generalizam exemplos por meio de outros e abstraem características (BRAGA; DE CARVALHO; LUDERMIR, 2000). Inicialmente as RNA foram inspiradas somente no funcionamento do cérebro humano e, posteriormente, foram introduzidos conceitos de estatística e processamento de sinais (BOTELHO, 2011);

b) **algoritmos genéticos:** são métodos computacionais adaptativos, baseados nos processos genéticos de organismos biológicos e podem ser usados para resolução de problemas de busca e otimização. Esses métodos utilizam conceitos como combinação genética, mutação e seleção natural, sendo úteis na resolução de problemas complexos que envolvem otimização, previsão e

simulação (SIVANANDAM; SUMATTI, 2006, tradução nossa);

c) **métodos estatísticos:** baseiam-se em princípios e teorias da estatística. Esses métodos fornecem modelos e técnicas tradicionais para análise e interpretação dos dados, como Redes Bayesianas[1], Análise Discriminante[2], Análise Exploratória de dados[3], entre outras técnicas disponíveis (GOLDSCHIMIDT; PASSOS, 2005);

d) **lógica *fuzzy:*** permite construir sistemas que lidem com informações imprecisas ou subjetivas. Enquanto os métodos baseados em lógica clássica permitem que um registro pertença a apenas um conjunto ou classe de dados, os métodos baseados em lógica *fuzzy* permitem que os registros pertençam a mais de uma classe simultaneamente (GOLDSCHIMIDT; PASSOS, 2005). Métodos baseados em lógica *fuzzy* são especialmente usados em clusterização de dados devido a sua capacidade em lidar com a imprecisão (BEZDEK, 2005, tradução nossa; PAL; MITRA, 2004, tradução nossa).

Goldschimidt e Passos (2005) exemplificam as diversas dificuldades decorrentes do processo de KDD, dentre as quais destacam a necessidade de manipulação de grandes e heterogêneos volumes de dados e a dificuldade de integração de vários algoritmos específicos.

Visando minimizar essas dificuldades, tem-se disponíveis ferramentas que implementam ambientes integrados para a realização de todo o processo de KDD. Essas

---

[1] A representação do conhecimento em sistemas especialista probabilísticos é realizada por meio de redes bayesianas. Uma rede bayesiana é um formalismo baseado na teoria dos grafos e na teoria da probabilidade total que possibilita a representação gráfica do conhecimento incerto e a propagação de probabilidades em sua estrutura por meio de algoritmos de inferência (CASTILHO; GUTIÉRREZ; HADI, 1998, tradução nossa).

[2] Técnica que pode ser utilizada para classificação de uma amostra ou população. Para sua realização é necessário que os grupos sejam conhecidos. Este conhecimento permite a elaboração de uma função matemática chamada de regra de discriminação, usada para classificar novos elementos amostrais nos grupos já existentes (MINGOTTI, 2005).

[3] Além da descrição dos dados, a análise exploratória permite que algumas características do processo sejam conhecidas, com base nos próprios dados. Faz uso de tabelas, gráficos e medidas estatísticas para tentar descobrir estrutura que não eram evidentes nos dados brutos (BARBETTA; REIS; BORNIA, 2010).

ferramentas são denominadas *shells* e visam facilitar a execução do processo de KDD.

Sabendo que a maior parte dessas ferramentas são comerciais, e visando ampliar o contingente de *shells* gratuitas a disposição da comunidade acadêmica e do público em geral, o Grupo de Pesquisa em Inteligência Computacional Aplicada, do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense (UNESC), formado por professores e acadêmicos do respectivo curso, mantém em desenvolvimento o projeto da *Shell Orion Data Mining Engine*.

2.1.2 *Shell Orion Data Mining Engine*

O projeto da *Shell Orion* foi iniciado no ano de 2005, e entre os principais objetivos do seu desenvolvimento está a disponibilização de uma ferramenta gratuita de DM para a comunidade em geral, sendo que os métodos implementados na *Shell Orion* são todos desenvolvidos por acadêmicos em seus respectivos Trabalhos de Conclusão de Curso (TCC).

A fim de demonstrar a constante evolução da ferramenta com a inserção de novas funcionalidades, na Tabela 1 estão sumarizados os algoritmos implementados até o momento na *Shell Orion*.

Tabela 1. Evolução da *Shell Orion Data Mining Engine*

Ano	Tarefa	Método	Algoritmo	Atributos	Referência
2005	Associação	Regras de Associação	Apriori	Numéricos	(CASAGRANDE, 2005)
2005	Classificação	Árvores de Decisão	ID3	Nominais	(PELEGRIN, 2005)
2007	Classificação	Árvores de Decisão	CART	Nominais e numéricos	(RAIMUNDO, 2007)
2007	Clusterização	Particionamento	*K-means*	Numéricos	(MARTINS, 2007)
2007	Clusterização	Redes Neurais	*Kohonen*	Numéricos	(BORTOLOTTO, 2007)
2008	Clusterização	Lógica *Fuzzy*	Gustafson-Kessel	Numéricos	(CASSETARI JUNIOR, 2008)
2009	Clusterização	Lógica *Fuzzy*	Gath-Geva	Numéricos	(PEREGO, 2009)
2009	Classificação	Árvores de Decisão	C4.5	Nominais e Numéricos	(MONDARDO, 2009)
2010	Classificação	Redes Neurais	RBF	Numéricos	(SCOTTI, 2010)
2010	Clusterização	Lógica *Fuzzy*	RCP	Numéricos	(CROTTI JUNIOR, 2010)
2010	Clusterização	Lógica *Fuzzy*	URCP	Numéricos	(CROTTI JUNIOR, 2010)
2010	Clusterização	Lógica Fuzzy	FCM	Numéricos	(CROTTI JUNIOR, 2010)

O desenvolvimento da *Shell Orion* é realizado na linguagem de programação Java, pois de acordo com Pelegrin (2005), essa linguagem permite reutilização de código, é independente de plataforma e possui ambientes de desenvolvimento gratuitos.

Outra vantagem da utilização da plataforma Java para desenvolvimento da *Shell Orion* é a sua Interface de Programação de Aplicações (*Application Programming Interface* - API) denominada *Java Database Connectivity* (JDBC). A utilização dessa API permite que a *Orion* se conecte a qualquer banco de dados que possua um *driver* disponível para ela, o que torna a ferramenta bastante flexível nesse sentido.

A fim de facilitar a interação do usuário com a ferramenta e permitir uma maior integração dos métodos disponibilizados, todas as funcionalidades da *Shell Orion* estão centralizadas em uma interface principal (Figura 3).

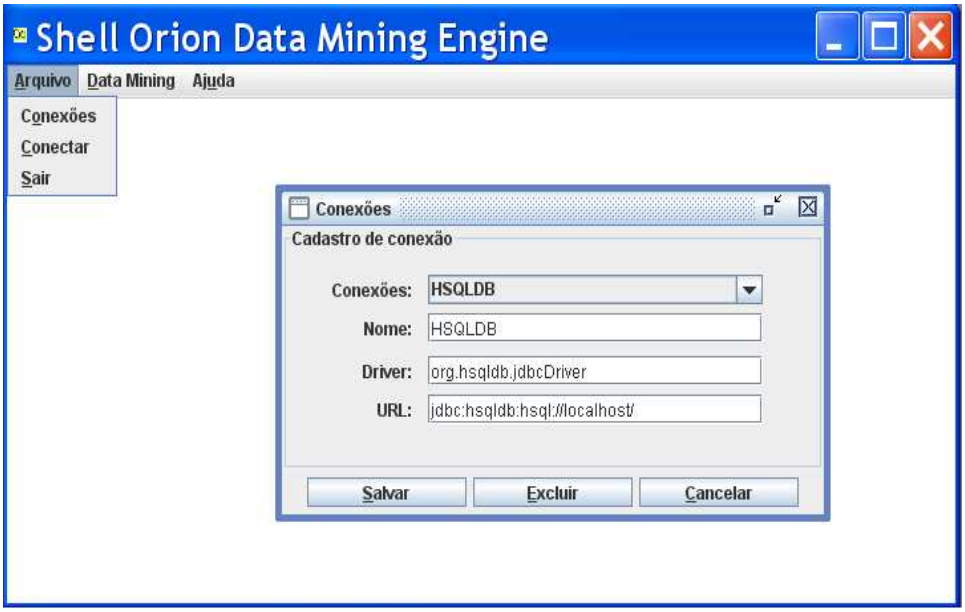

Figura 3. Interface principal da *Shell Orion Data Mining Engine*

Após ser realizada a conexão com uma base de dados previamente cadastrada, pode-se acessar o item do menu *Data Mining*, onde se tem acesso às tarefas e aos métodos disponibilizados na ferramenta.

Atualmente a *Shell Orion* está organizada em módulos diferentes para cada tarefa de DM, possuindo métodos que contemplam as tarefas de classificação, clusterização e associação. Cada algoritmo implementado é independente dos demais, necessitando de informações específicas para funcionarem adequadamente, de acordo com a tarefa que se deseja realizar.

Considerando-se que o projeto *Shell Orion* encontra-se em desenvolvimento desde 2005 e que vários Trabalhos de Conclusão de Curso já abordaram sobre esta ferramenta, no Apêndice A encontram-se informações mais detalhadas acerca de seus diferentes módulos, como por exemplo, o de clusterização.

A tarefa de clusterização da *Shell Orion* possui diversos algoritmos implementados. O algoritmo DBSCAN, foco dessa pesquisa, visa ampliar as funcionalidades da ferramenta, disponibilizando o método de densidade no módulo de clusterização.

3 A TAREFA DE CLUSTERIZAÇÃO EM *DATA MINING*

A tarefa de segmentação de um grupo heterogêneo de dados em vários subgrupos, também chamados de *clusters*, é denominada clusterização. Diferentemente da tarefa de classificação, na clusterização não existem classes pré-definidas ou exemplos, sendo que os registros são agrupados conforme a sua similaridade com os demais dados (BERRY; LINOFF, 2004, tradução nossa). Portanto, o principal objetivo da tarefa de clusterização é procurar por uma estrutura conveniente e válida em um conjunto de dados, e não estabelecer regras de separação dos registros em categorias pré-definidas (JAIN; DUBES, 1988, tradução nossa).

De acordo com Larose (2005) um *cluster* pode ser entendido como uma coleção de registros que são similares entre si e dissimilares de objetos em outros *clusters*, ou seja, objetos pertencentes a um dado *cluster* devem compartilhar um conjunto de propriedades comuns, sendo que essas propriedades não são compartilhadas com objetos de outros *clusters* (GOLDSCHIMIDT; PASSOS, 2005).

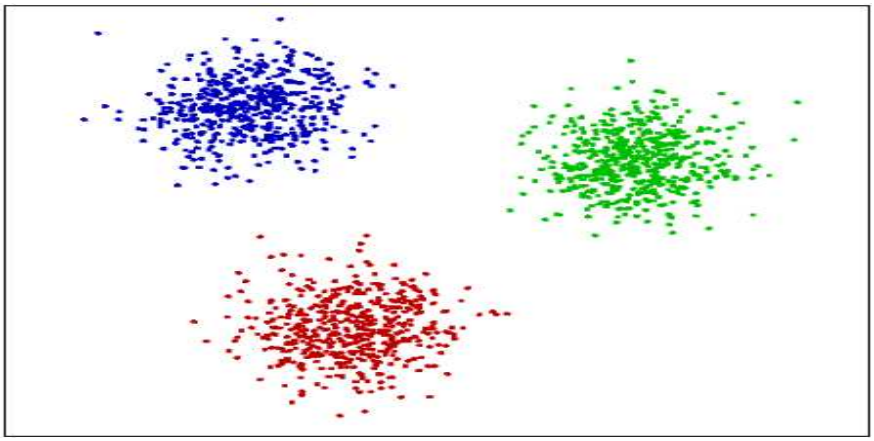

Figura 4. Exemplo de um conjunto de dados agrupados em três *clusters*
Fonte: PATERLINI, A. (2011).

A divisão do conjunto inicial de dados resultante do processo de clusterização pode ser usada de duas maneiras distintas. Ora para produzir uma sumarização da base de

dados em questão por meio das características de cada *cluster*, ora como dados de entrada para outras técnicas, como por exemplo, a classificação, que trabalharia em cima dos resultados obtidos pela tarefa de clusterização (SASSI, 2006).

Conforme Jain et al (1999) existem alguns fatores que devem ser levados em consideração durante a tarefa de clusterização: a seleção e a preparação dos dados, a medida de similaridade adequada, o algoritmo adotado e a validação dos resultados gerados. A abordagem que é dada a cada um desses fatores é determinante no resultado final do processo, influenciando na qualidade da divisão dos *clusters* (OLIVEIRA, 2008).

A fim de atingir o objetivo proposto, a tarefa de clusterização apresenta várias etapas (Figura 5) que vão desde a preparação dos objetos até a interpretação dos *clusters* obtidos (FACELI, 2007; JAIN et al,1999, tradução nossa; NALDI, 2011):

a) **preparação dos dados:** os dados que serão submetidos a um algoritmo de clusterização devem estar padronizados, portanto, a etapa de preparação dos dados envolve vários aspectos relacionados ao seu pré-processamento e a forma de representação apropriada para sua utilização por um algoritmo de clusterização (FACELI, 2007). Nessa etapa podem ocorrer normalizações, conversões de tipos e redução do número de atributos por meio de seleção ou extração do número de características (JAIN; MURTY; FLYNN, 1999, tradução nossa);

b) **medida de similaridade/dissimilaridade:** a escolha de uma medida de similaridade ou de dissimilaridade é uma importante etapa na tarefa de clusterização. Na primeira, quanto maior o valor observado, mais parecidos entre si serão os objetos. Já para a segunda, quanto maior o valor observado, menos parecidos serão os objetos (CARVALHO, 2005). A escolha de uma medida deve levar em conta os tipos e escalas dos atributos que definem os

objetos e as propriedades que serão focadas durante a tarefa (JAIN; MURTY; FLYNN, 1999, tradução nossa). Na maioria dos casos, é preciso garantir que todos os atributos contribuam igualmente para o cálculo da medida (GUNOPULOS, 2009, tradução nossa);

c) **realização do agrupamento:** são aplicados os algoritmos de clusterização apropriados para agrupar os dados de acordo com o objetivo específico (NALDI, 2011). Os algoritmos de clusterização baseiam-se em duas idéias: coesão interna dos objetos e isolamento externo entre os grupos, sendo que todos os algoritmos tentam maximizar as diferenças entre grupos relativas à variação dentro dos grupos (CARVALHO, 2005). Os *clusters* resultantes podem ser exclusivos (*crisp*), onde um objeto pertence ou não pertence ao *cluster*, ou podem ser não-exclusivos (*fuzzy*), onde um objeto pertence a um grupo com determinado grau de pertinência, podendo assim pertencer a mais de um grupo ao mesmo tempo (JAIN; MURTY; FLYNN, 1999, tradução nossa);

d) **validação:** faz referência aos procedimentos de avaliação dos resultados obtidos de forma quantitativa e objetiva (JAIN; DUBES, 1988, tradução nossa). Deve ser determinado de forma objetiva se os *clusters* encontrados são significativos, ou seja, se a solução é representativa para o conjunto de dados analisado (FACELI, 2007). Geralmente essa validação é feita com base em índices estatísticos, que quantificam alguma informação a respeito da qualidade de um *cluster*, ou estimam o grau com que a estrutura resultante reflete o conjunto de dados original (NALDI, 2011);

e) **interpretação:** nesta etapa, os *clusters* resultantes são avaliados, em relação aos seus padrões observados, com o objetivo de descrever a natureza de cada

cluster (NALDI, 2011). Além de ser uma forma de avaliação dos *clusters* encontrados e da hipótese inicial, de um modo confirmatório, os *clusters* podem permitir avaliações subjetivas que tenham um significado prático (FACELI, 2007). Dependendo do objetivo da tarefa, essa etapa não é realizada, sendo que quando ela ocorre, é realizada por um especialista no domínio de aplicação. Alguns algoritmos podem realizar a clusterização sem que seja necessário informar o número de *clusters* desejados antecipadamente, sendo que nesses casos a partição final do conjunto de dados normalmente requer alguma avaliação. Em alguns casos, os especialistas precisam acrescentar outras evidências experimentais e analíticas para chegar a uma conclusão objetiva do resultado (GUNOPULOS, 2009, tradução nossa).

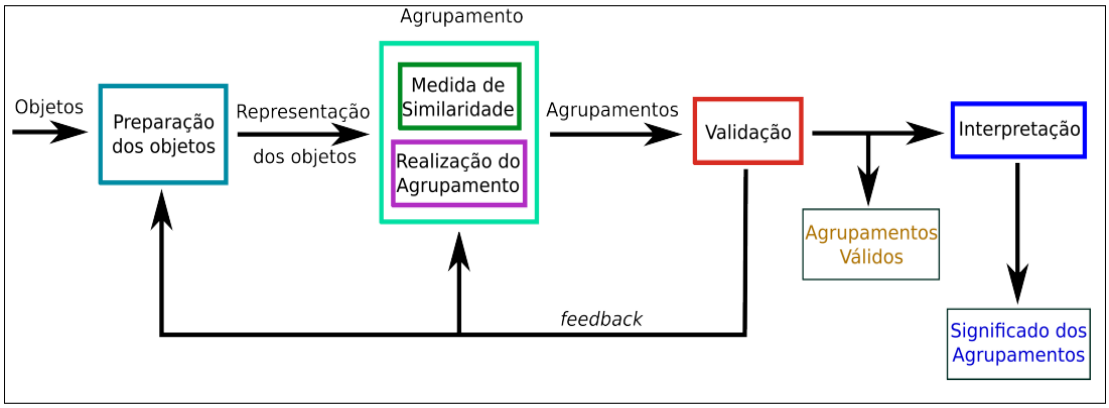

Figura 5. Etapas do processo de clusterização
Fonte: Adaptado de NALDI, M. (2011)

A tarefa de clusterização é subjetiva, pois o mesmo conjunto de dados muitas vezes precisa ser dividido de diferentes formas para aplicações distintas, existindo diversos métodos de clusterização propostos. Visto que cada método emprega um critério de agrupamento que impõe uma estrutura nos dados e possui uma complexidade computacional particular, os métodos são usados de acordo com as características do conjunto de dados, conforme o conhecimento do domínio de aplicação que os usuários dispõem e também conforme o resultado desejado (JAIN;

MURTY; FLYNN, 1999, tradução nossa).

Com a finalidade de facilitar a compreensão e a implementação dos algoritmos de clusterização, os diferentes métodos de clusterização podem ser classificados da seguinte maneira (JAIN; MURTY; FLYNN, 1999, tradução nossa):

a) **métodos hierárquicos:** o método de clusterização hierárquico cria uma decomposição do conjunto de dados, que pode ser representada por um dendograma[4] (GOLDSCHIMIDT; PASSOS, 2005). Baseado na maneira em que a decomposição é formada, os métodos hierárquicos podem ser classificados em duas abordagens diferentes: aglomerativa (*bottom-up*)[5] e divisiva (*top-down*)[6] (JAIN; DUBES, 1988, tradução nossa; WITTEN; FRANK; HALL, 2011, tradução nossa);

b) **métodos de particionamento:** dividem a base de dados em *k* grupos, onde esse número *k* é dado pelo usuário. Nesse tipo de método, é pressuposto que o usuário tenha conhecimento de quantos *clusters* existem no conjunto de dados (CARVALHO, 2005; PAL; MITRA, 2004, tradução nossa). Métodos de particionamento obedecem à premissa de que uma partição deve conter ao menos um objeto e cada um deve pertencer somente a uma partição, sendo que esse requerimento pode ser minimizado com o uso de técnicas de lógica *fuzzy*, em que um objeto pertence a um grupo com determinado grau de

[4] Árvore que iterativamente decompõe o conjunto de objetos em subconjuntos menores até que cada um desses subconjuntos consista de somente um objeto (ESTER et al,1996,tradução nossa).

[5] Inicialmente cada objeto forma um *cluster* separado. Então, através de sucessivas iterações, pares desses *clusters* são agrupados conforme a medida de distância entre eles, sendo que essas distâncias geralmente estão agrupadas em uma matriz de distância simétrica. O algoritmo *Aglomerative Nesting* (AGNES) é um exemplo de algoritmo que usa o método hierárquico aglomerativo (CARLANTONIO, 2001; HAN; KAMBER, 2006, tradução nossa).

[6] Inicialmente todos os objetos estão em um único *cluster*. Então, em sucessivas iterações, o *cluster* inicial é subdivido de acordo com a dissimilaridade entre os objetos e são feitos *clusters* cada vez menores. Esse processo continua até que cada objeto represente um *cluster* ou uma condição de termino, como o número de *clusters* desejados, seja atingida. O algoritmo *Divisive Analysis* (DIANA) é um representante dessa categoria de algoritmos hierárquicos (CARVALHO, 2005; HAN; KAMBER, 2006, tradução nossa; PAL; MITRA, 2004, tradução nossa).

pertinência (HAN; KAMBER, 2006, tradução nossa; JAIN; MURTY; FLYNN, 1999, tradução nossa). Os algoritmos *k-means*[7] e *k-medoids*[8] são representantes do método de particionamento;

c) **métodos baseados em grade:** dividem o espaço dos dados em um número finito de células que formam uma estrutura de grade, no qual todas as operações de clusterização são executadas (GAN; MA; WU, 2007, tradução nossa). O STING[9] é um exemplo de algoritmo que utiliza a metodologia baseada em grades;

d) **métodos baseados em modelos:** utilizam modelos matemáticos para definir os *clusters* e estruturar o conjunto de dados. Os modelos são construídos, baseando-se na idéia de que os dados são gerados por uma mistura de distribuição de probabilidades (GAN; MA; WU, 2007, tradução nossa). Geralmente, os algoritmos que utilizam modelos são construídos utilizando duas abordagens distintas (HAN; KAMBER, 2006, tradução nossa): estatística[10] e redes neurais[11]. Os algoritmos COBWEB[12] e *Konohen*[13] são considerados exemplos dessa abordagem;

e) **métodos baseados em densidade:** nesse tipo de método, um *cluster* é

---

[7] Cada agrupamento é representado por um centro, que é calculado pela média (ou média ponderada) dos elementos que o compõem. Esse cálculo de média gera o chamado centro de gravidade do *cluster* (GOLDSCHIMIDT; PASSOS, 2005; PATERLINI, 2011).

[8] Essa estratégia toma como representante do agrupamento o objeto que estiver mais próximo do centro de gravidade do mesmo, sendo esse elemento denominado *medoid.* O elemento mais central do *cluster* será aquele que minimiza a soma das distâncias para todos os outros elementos (PATERLINI, 2011).

[9] No algoritmo STING, a área espacial é dividida em células retangulares, existindo diversos níveis dessas células, que formam uma estrutura hierárquica, onde cada célula no nível mais alto é particionada para formar um número de células no próximo nível mais baixo (HAN; KAMBER, 2006, tradução nossa).

[10] Primeiramente é realizada a clusterização convencional, e após isso é realizada uma etapa adicional, onde para cada agrupamento são encontradas descrições características para cada grupo, que ira representar um conceito ou classe (HAN; KAMBER, 2006, tradução nossa).

[11] Cada *cluster* é considerado um exemplar e então novos objetos podem ser distribuídos para *clusters* cujo exemplar é o mais similar, de acordo com alguma medida de similaridade (HAN; KAMBER, 2006, tradução nossa).

[12] O algoritmo COBWEB recebe como parâmetro de entrada pares de valores de atributos categóricos, e cria uma hierarquia de *clusters* na forma de árvores de decisão (HAN; KAMBER, 2006, tradução nossa).

[13] Organiza dimensionalmente dados complexos em *clusters*, baseado em suas relações, de tal forma que objetos similares estejam próximos um do outro (GAN; MA; WU, 2007, tradução nossa).

considerado uma região em que a densidade de elementos excede certo limiar, ou seja, para cada elemento de um dado agrupamento, sua vizinhança, em um certo raio, deve conter ao menos uma quantidade mínima de elementos (HAN; KAMBER, 2006, tradução nossa; PATERLINI, 2011). O algoritmo DBSCAN utiliza a abordagem baseada em densidade.

Na Tabela 2 é possível verificar de maneira sumarizada algumas das principais vantagens e desvantagens dos diversos métodos de clusterização existentes (ESTER et al, 1996 tradução nossa; HAN; KAMBER, 2006, tradução nossa; JAIN; MURTY; FLYNN, 1999, tradução nossa; METZ, 2006; PAL; MITRA, 2004, tradução nossa).

Tabela 2. Métodos de clusterização

Método	Vantagens	Desvantagens
Hierárquico	• liberdade em relação ao nível de granularidade desejado • uso de qualquer medida de similaridade	• definição dos parâmetros de parada dos algoritmos • dificuldade de representação dos agrupamentos criados
Particionamento	• boa eficiência computacional • efetivo se o número de *clusters* puder ser estimado com antecedência	• definição do número de partições que devem ser criadas • dificuldades em trabalhar com *clusters* com formato arbitrário e conjuntos de dados contendo *outliers*
Grade	• robustez na presença de *outliers* • encontra *clusters* com formato arbitrário • tempo de processamento independente do tamanho da base de dados	• tamanho das células em que o espaço dos dados é dividido afeta o resultado da clusterização • dificuldade em determinar os parâmetros de entrada
Modelos	• capacidade de identificação de *outliers* • identificação automática do número de *clusters*	• complexidade computacional elevada
Densidade	• forma grupos com formatos arbitrários • identifica *outliers* • uso de qualquer medida de similaridade	• alta complexidade computacional • não trabalha bem com conjuntos de dados multidimensionais

Grande parte dos métodos de clusterização geralmente encontra dificuldades para gerar *clusters* com formatos arbitrários, pois criam agrupamentos baseados somente nas

medidas de distância entre os objetos, e não retornam bons resultados quando o conjunto de dados em questão possui *outliers* (ESTER et al, 1996, tradução nossa).

Visando suprir as deficiências acima expostas, e buscando diminuir a estrutura que a maioria dos métodos de clusterização impõe ao conjunto de dados, foi proposto o método de clusterização baseado em densidade (HAN; KAMBER, 2006, tradução nossa).

3.1 O MÉTODO DE DENSIDADE

Algoritmos de clusterização de dados baseados em densidade são usados para a descoberta de agrupamentos de forma arbitrária, especialmente em conjuntos de dados contendo *outliers* (PATERLINI, 2011). Esses algoritmos encontram agrupamentos baseados na densidade de elementos em uma determinada região no espaço de objetos.

De acordo com o método baseado em densidade, um *cluster* pode ser definido como uma região densa no espaço de dados, que é separada de outras áreas densas, por regiões de baixa densidade, que representam *outliers*. Essas áreas podem ter um formato arbitrário e ainda os pontos dentro de uma região podem estar distribuídos arbitrariamente (ANKERST et al, 1999, tradução nossa; CARLANTONIO, 2001; HAN; KAMBER, 2006, tradução nossa; PATERLINI, 2011).

*Cluster*s são reconhecidos principalmente porque dentro de cada um deles, a densidade de objetos é maior do que a densidade de objetos fora do grupo. Por exemplo, na Figura 6 em (a) é possível verificar que existem quatro *clusters* de tamanhos distintos, mais com formato arredondado, (b) mostra quatro grupos com formato arbitrário e de tamanho diverso. Por fim, em (c) é possível verificar a existência de quatro agrupamentos com formato arbitrário, densidade distinta e diversos pontos dispersos fora dos *clusters*, que representam *outliers*.

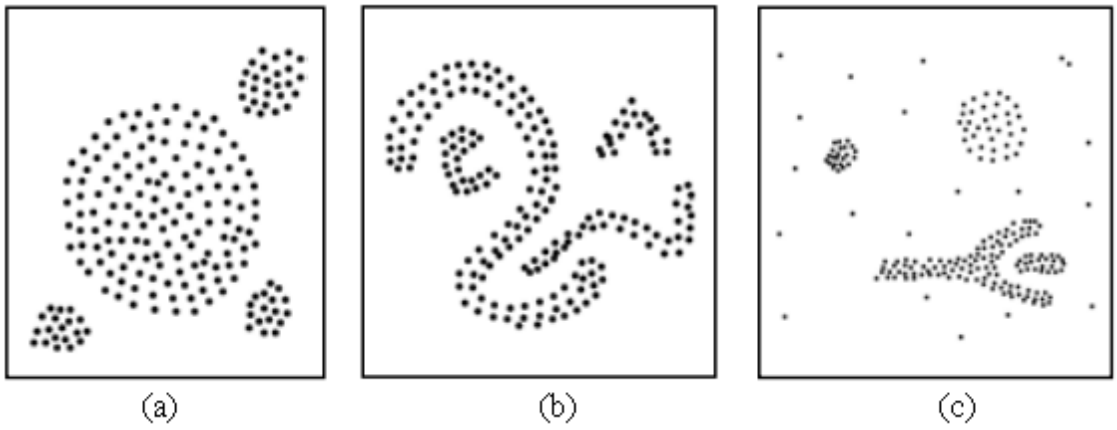

Figura 6. *Clusters* com diferentes tamanhos, formatos e densidades
Fonte: Adaptado de ESTER, M. et al (1996)

O critério de clusterização local é utilizado por algoritmos que implementam o método de densidade, pois esses algoritmos consideram densidade de ligações entre os dados no espaço métrico. Considerando que um objeto com *n* atributos pode ser representado como um ponto em um espaço *d*-dimensional, então os *clusters* correspondem a subconjuntos de objetos que estejam próximos. Assim, agrupamentos localizam-se em regiões de maior densidade no espaço métrico e são separados por regiões de baixa densidade (ANKERST et al, 1999, tradução nossa; OLIVEIRA, 2008).

Métodos baseados em densidade são adequados para encontrar agrupamentos com vários formatos distintos, pois a forma dos grupos é determinada pela medida de distância escolhida e esse método tem a capacidade de trabalhar com diversas dessas medidas. Por exemplo, se a distância *Manhattan*[14] é utilizada em um espaço bidimensional, o formato dos agrupamentos tende a ser retangular. Portanto, a escolha da função de distância deve ser feita de acordo com cada aplicação em particular e também de acordo com o tipo de dado que será utilizado (ESTER et al,1996, tradução nossa).

Esses métodos podem formar *clusters* de acordo com um centro de densidade ou conforme alguma função de distribuição de densidade, existindo diversos algoritmos, sendo

[14] A função de distância *Manhattan*, também denominada *City-Block*, corresponde ao somatório do módulo das diferenças dos valores dos atributos entre dois pontos (KAUFMAN; ROUSSEEAW, 1990, tradução nossa)

que cada um utiliza uma técnica distinta, de acordo com o problema que se propõe a resolver (CARLANTONIO, 2001).

A abordagem baseada em centro de densidade segue conceitos de conectividade e alcançabilidade, em que cada ponto tem relação com os vizinhos mais próximos e os *clusters* crescem na direção em que a densidade de pontos dentro dos grupos indicarem. Já a abordagem baseada em funções de distribuição de densidade utiliza modelos matemáticos para determinar a influência que cada ponto exerce em sua vizinhança de objetos, gerando *clusters* de acordo com os pontos que exercem maior uma maior influência nos seus vizinhos próximos (HAN; KAMBER, 2006, tradução nossa; HINNEBURG; KEIM, 1998, tradução nossa).

Existem diversos algoritmos que utilizam o método de clusterização baseada em densidade, sendo que cada um possui seus conceitos particulares para definir áreas com alta densidade de objetos no espaço dos dados. Alguns algoritmos que podem ser destacados são:

a) **DBSCAN:** proposto por Ester et al (1996), o DBSCAN é considerado referência entre os algoritmos que usam a abordagem baseada em densidade. Este algoritmo foi desenvolvido para ser aplicado em grandes bases de dados contaminadas por *outliers* e busca minimizar a necessidade de conhecimento prévio do conjunto de dados ao mesmo tempo em que encontra *clusters* com diversos formatos e com eficiência aceitável (ESTER et al, 1996, tradução nossa). O DBSCAN segue a abordagem baseada em centro de densidade, sendo que a densidade de pontos no espaço dos dados é estimada pela contagem dos pontos contidos dentro de um determinado raio de vizinhança a partir de um ponto do conjunto de dados, que deve conter um número mínimo de pontos (CARLANTONIO, 2001; ESTER et al, 1996, tradução nossa);
b) **OPTICS:** o algoritmo *Ordering Points to Identify the Clustering Structure*

(OPTICS) foi proposto por Ankerst et al (1999) e estende o algoritmo DBSCAN para que vários valores de distância sejam processados simultaneamente, construindo diversos agrupamentos com densidades diferentes ao mesmo tempo (HAN; KAMBER, 2006, tradução nossa). Utilizando um valor global para estimar a densidade no espaço dos dados, agrupamentos com alta densidade podem ser completamente contidos em grupos menos compactos. Para resolver esse tipo de problema, o algoritmo o OPTICS processa simultaneamente vários valores de distância, construindo diversos *clusters* com densidades distintas ao mesmo tempo (HAN; KAMBER, 2006, tradução nossa). Para a construção de *clusters* simultâneos, os pontos devem ser processados em uma ordem específica, sendo que os *clusters* mais densos são encerrados primeiro. Então o OPTICS produz uma ordenação do conjunto de dados, de modo que o resultado do agrupamento possa ser facilmente visualizado e computado (ANKERST et al, 1999, tradução nossa);

c) **DENCLUE:** o algoritmo *Density-Based Clustering* (DENCLUE) foi proposto por Hinneburg e Keim (1998) sendo baseado em um conjunto de funções de distribuição de densidade, tendo como proposta ser eficiente em bases de dados com forte presença de *outliers*. A idéia básica desse algoritmo diz que a influência de cada ponto em sua vizinhança pode ser modelada matematicamente por meio de uma função de influência[15] (HINNEBURG; KEIM, 1998, tradução nossa). A densidade global do conjunto de dados é modelada analiticamente como a soma das funções de influência de todos os

[15] A influência de cada ponto de um conjunto de dados pode ser formalmente modelada por meio de uma função matemática, chamada de função de influência, que descreve o impacto que um ponto exerce em seus objetos vizinhos. Em principio, a função de influência pode ser uma função arbitraria determinada pela distância entre dois objetos vizinhos, tal como a função de influência Gaussiana (HINNEBURG; KEIM, 1998, tradução nossa).

pontos do conjunto de objetos, sendo que os *clusters* podem ser determinados pela identificação dos atratores de densidade, onde esses atratores são máximos locais da função de densidade global (HAN; KAMBER, 2006, tradução nossa; HINNEBURG; KEIM, 1998, tradução nossa);

d) **SNN:** o algoritmo *Shared Nearest Neighbor* (SNN) foi proposto por Ertöz, Steinbach e Kumar (2003) e tem como critério de agrupamento o encadeamento ou ligação entre os pontos que serão agrupados. O SNN encontra os vizinhos mais próximos de cada ponto, utilizando como medida de proximidade o número de vizinhos que dois pontos compartilham no espaço dos dados (ERTÖZ; STEINBACH; KUMAR, 2003, tradução nossa). Com essa medida de proximidade, o algoritmo procura pelos pontos mais representativos, construindo grupos ao redor desses objetos (ERTÖZ; STEINBACH; KUMAR, 2003, tradução nossa);

e) **CLIQUE:** proposto por Agrawal et al (1998) o algoritmo *Clustering in Quest* (CLIQUE) se baseia em grades e densidade, sendo um algoritmo misto que particiona o conjunto de dados em células com a finalidade de encontrar agrupamentos compactos. Baseado na idéia de que o espaço de dados é ocupado de maneira não uniforme, o CLIQUE distingue áreas densas de regiões com escassez de pontos, encontrando os padrões de distribuição de densidade do conjunto. Uma área é considerada densa se a quantidade de pontos contidos nesse local excede um dado parâmetro de entrada, sendo que *clusters* são formados realizando a junção de áreas adjacentes (AGRAWAL et al, 1998, tradução nossa; GAN; MA; WU, 2007, tradução nossa).

Na Tabela 3 estão expostas as principais vantagens e desvantagens de alguns algoritmos de clusterização que utilizam conceitos de densidade para a tarefa de

clusterização (ERTÖZ; STEINBACH; KUMAR, 2003, tradução nossa; ESTER et al, 1996, tradução nossa; GAN; MA; WU, 2007, tradução nossa; HAN; KAMBER, 2006, tradução nossa):

Tabela 3- Algoritmos de clusterização baseados em densidade

Algoritmo	Vantagens	Desvantagens
DBSCAN	• robustez na presença de *outliers* • encontra *clusters* com formato arbitrário • trabalha com várias medidas de distância	• definições dos parâmetros de densidade são empíricos e de difícil determinação • não trabalha bem quando *clusters* possuem densidades muito distintas
OPTICS	• não se limita a um único parâmetro de densidade • robustez na presença de *outliers* • técnicas de visualização podem auxiliar no conhecimento da distribuição dos dados	• complexidade computacional elevada • não trabalha bem com conjuntos de dados multidimensionais
DENCLUE	• sólida fundamentação matemática • robustez na presença de *outliers* • eficiência para trabalha com grandes conjuntos de dados	• requer cuidado na definição dos parâmetros de entrada • não trabalha bem com conjuntos de dados multidimensionais
SNN	• robustez na presença de *outliers* • não é tão afetado por conjuntos de dados com alta dimensionalidade • pode encontrar *clusters* com densidades variadas	• complexidade computacional elevada • grande número de parâmetros de entrada necessários ao algoritmo
CLIQUE	• não é tão afetado por conjuntos de dados com alta dimensionalidade • relativa simplicidade de implementação • insensível a ordem de entrada dos dados	• precisão do resultado pode ser degradada pela simplicidade do método • parâmetro de densidade constante, mesmo com o aumento da dimensionalidade

Considerando que o objetivo geral dessa pesquisa consiste na implementação do algoritmo DBSCAN, suas características e funcionalidades são descritas a seguir, com a finalidade de possibilitar o seu entendimento.

4 O ALGORITMO DBSCAN

O algoritmo *Density Based Spatial Clustering of Applications With Noise* (DBSCAN) foi proposto por Martin Ester, Hans-Peter Kriegel, Jörg Sander e Xiaowei Xu na *Second International Conference on Knowledge Discovery and Data Mining* no ano de 1996 na cidade de Portland nos Estados Unidos.

Os grupos formados pelo algoritmo são compostos por objetos de borda e centrais, que estão ligados entre si por alguma medida de similaridade. O algoritmo se baseia no critério de encadeamento, onde objetos vizinhos devem compartilhar o mesmo *cluster*, sendo esse critério adequado para a detecção de *clusters* com formas arbitrárias.

Um *cluster* é formado por um conjunto de pontos conectados por densidade, de acordo com os parâmetros de entrada do algoritmo, ou seja, após ser informado o tamanho do raio de vizinhança e o número mínimo de pontos que devem estar contidos nesse raio, o algoritmo iterativamente recupera a vizinhança de cada objeto da base de dados, procurando por regiões em que o limiar de densidade é excedido, permitindo assim que um *cluster* possa ser formado em uma região densa de acordo com os parâmetros de entrada informados (GAN; MA; WU, 2007, tradução nossa; HAN; KAMBER, 2006, tradução nossa).

Para cada novo objeto adicionado a um grupo sua densidade é calculada e assim novos objetos são adicionados, sendo que dessa forma o *cluster* vai crescendo de acordo com a densidade de ligações entre os pontos, assim podendo assumir um formato arbitrário (FACELLI, 2006; NALDI, 2011; PAL; MITRA, 2004, tradução nossa).

Algumas definições são necessárias para o entendimento do algoritmo DBSCAN, e da noção de densidade de ligação entre objetos, utilizada pelo algoritmo.

4.1 O RAIO DE VIZINHANÇA DE UM PONTO

O DBSCAN trabalha com a idéia de que para cada objeto de um *cluster*, sua vizinhança, para algum dado raio ε (épsilon), deve conter ao menos um número mínimo de pontos (η) para ser considerado um *cluster*, onde ε e η são parâmetros de entrada para o algoritmo (ESTER et al,1998, tradução nossa; PATERLINI, 2011).

A ε-vizinhança de um ponto x_p, que é a vizinhança de um objeto em um dado raio ε, denotada por $N_\varepsilon(x_p)$, é definida por:

$$N_\varepsilon(x_p) = \{x_q \in D \mid dist(x_p, x_q) \leq \varepsilon\}$$

Se a distância entre um ponto x_p e um ponto x_q for menor ou igual a ε, ou seja, $dist(x_p, x_q) \leq \varepsilon$, então o ponto x_q está na ε-vizinhança do ponto x_p.

Em um conjunto de dados podem ser definidos três tipos de pontos:

a) **pontos centrais:** são pontos que estão no interior de uma região densa, onde existem pelo menos η pontos no raio ε desse objeto. A cardinalidade[16] (*Card*) desses pontos em relação ao parâmetro ε deve ser de no mínimo η pontos, podendo ser denotada pela seguinte definição (ANKERST et al,1999,tradução nossa);

$$Card(N_\varepsilon(x_q)) \geq \eta;$$

b) **pontos de borda:** estão na fronteira de uma região densa, ou seja, são pontos que estão na ε-vizinhança de algum ponto central, porém não são pontos centrais, pois a cardinalidade desses pontos em relação ao raio ε não excede η (ANKERST et al,1999,tradução nossa);

$$Card(N_\varepsilon(x_q)) \leq \eta;$$

[16] Definida como o número de elementos que pertencem a um determinado conjunto. Por exemplo, seja o conjunto A = {-1, 0, 1, 2, 3}, então a cardinalidade desse pode ser definida por *Card*(A) = 5 (GERSTING, 1995).

c) ***outliers*:** esses pontos não são centrais e nem de borda e assim não são conectados por densidade a nenhum outro ponto, não pertencendo a nenhum *cluster*.

Não deve ser exigido que todos os pontos de um *cluster* contenham o número mínimo de pontos η em sua ε-vizinhança, pois geralmente os pontos de borda de um *cluster* possuem um número menor de pontos em sua ε-vizinhança se comparados a pontos centrais.

Porém, é necessário que para cada ponto x_p em um cluster *C,* exista um ponto x_q nesse mesmo *cluster C* de modo que x_p esteja dentro da ε-vizinhança de x_q . A $N_\varepsilon(x_q)$ por sua vez deve possuir pelo menos η pontos, ou seja, a cardinalidade de x_q em relação ao raio ε deve exceder η. Assim, todos os pontos em um grupo são alcançáveis por densidade a partir de qualquer ponto central do *cluster* (ESTER et al, 1996, tradução nossa).

4. 2 PONTOS DIRETAMENTE ALCANÇÁVEIS POR DENSIDADE

Um ponto x_p é diretamente alcançável por densidade a partir de um ponto x_q se as seguintes condições forem satisfeitas (ESTER et al,1996, tradução nossa):

$$x_p \in N_\varepsilon(x_q)$$

$$N_\varepsilon(x_q) \geq \eta$$

Ou seja, o ponto x_p deve estar contido na ε-vizinhança do ponto x_q e a ε-vizinhança do ponto x_q deve exceder ou ser igual à η, sendo que a cardinalidade do ponto x_q com relação ao raio ε deve ser de no mínimo η, satisfazendo assim a sua condição de objeto central em um *cluster* (ESTER et al,1996,tradução nossa; KRIEGEL et al, 2011, tradução nossa).

Essa é uma propriedade simétrica para pares de objetos centrais, pois um objeto central é diretamente alcançável por densidade a partir de outro ponto central e vice-versa,

porém, a relação é quase sempre assimétrica quando estão envolvidos um ponto central e um ponto de borda em um mesmo *cluster* (ESTER et al,1996,tradução nossa). Na Figura 7 é possível verificar que o ponto *p* é diretamente alcançável por densidade a partir do ponto *q*, pois *q* é um ponto central, porém, *q* não é diretamente alcançável por densidade a partir de *p*, porque apesar de *q* estar contido na *ε*-vizinhança de *p*, esse ponto não satisfaz a condição de ponto central, pois não possui o número mínimo de pontos vizinhos necessários para isso.

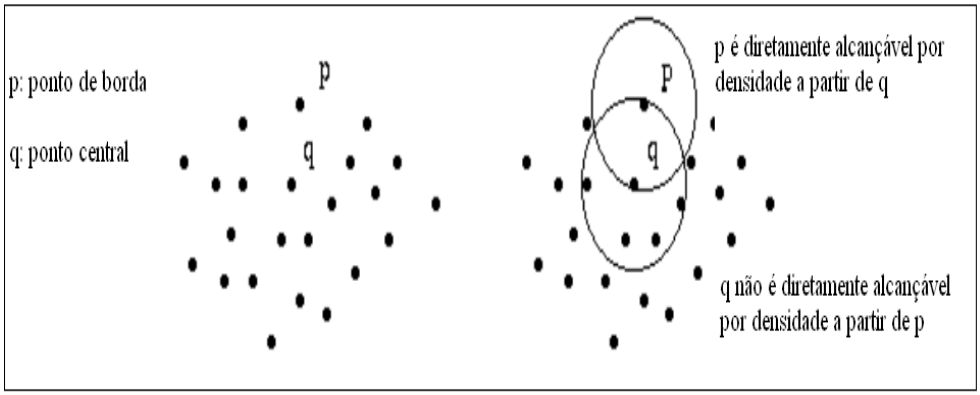

Figura 7. Pontos de borda e pontos centrais
Fonte: Adaptado de ESTER, M. et al (1996)

A propriedade que diz respeito aos pontos diretamente alcançáveis por densidade pode ser estendida para gerar uma nova definição, chamada de pontos indiretamente alcançáveis por densidade.

4.3 PONTOS INDIRETAMENTE ALCANÇÁVEIS POR DENSIDADE

Um ponto x_p é indiretamente alcançável por densidade a partir de um ponto x_q, levando em consideração os valores *ε* e *η*, se existir uma cadeia de pontos $x_{p_1},\ldots, x_{p_n}$, tal que $x_{p_1} = x_q$ e $x_{p_n} = x_q$, em que $x_{p_{i+1}}$ é diretamente alcançável por densidade a partir de x_{p_i} (ESTER et al, 1996, tradução nossa; HAN; KAMBER, 2011, tradução nossa).

Pontos indiretamente alcançáveis por densidade possuem uma relação transitiva

que se estabelece entre três elementos de um mesmo grupo, de tal forma que se o primeiro elemento tem relação com o segundo e este detém uma relação com o terceiro, então o primeiro elemento tem relação com o terceiro (ESTER et al, 1996, tradução nossa; GERSTING, 1995).

Apesar da transitividade, pontos indiretamente alcançáveis por densidade somente terão uma relação simétrica caso estiverem envolvidos nessa relação dois pontos centrais de um *cluster* (ESTER et al, 1996, tradução nossa).

Portanto, a relação entre pontos indiretamente alcançáveis por densidade é estendida para pontos de borda, gerando o conceito de pontos conectados por densidade.

4.4 PONTOS CONECTADOS POR DENSIDADE

Dois pontos de borda em um *cluster*, não são indiretamente alcançáveis por densidade entre si, pois os mesmos não detêm a condição de pontos centrais. Contudo, deve existir um ponto central no *cluster* a partir do qual esses pontos de borda são indiretamente alcançáveis por densidade, assim, sendo conectados por densidade entre si (ESTER et al, 1996, tradução nossa).

Um ponto x_p é conectado por densidade a um ponto x_q, se existir um ponto x_z de tal forma que, ambos os pontos x_p e x_q são indiretamente alcançáveis por densidade a partir de x_z (ESTER et al, 1996, tradução nossa; KREGEL et al, 2011, tradução nossa).

A relação entre pontos conectados por densidade é considerada simétrica. A Figura 8 demonstra as definições apresentadas e mostra as diferenças entre pontos indiretamente alcançáveis por densidade e pontos conectados por densidade.

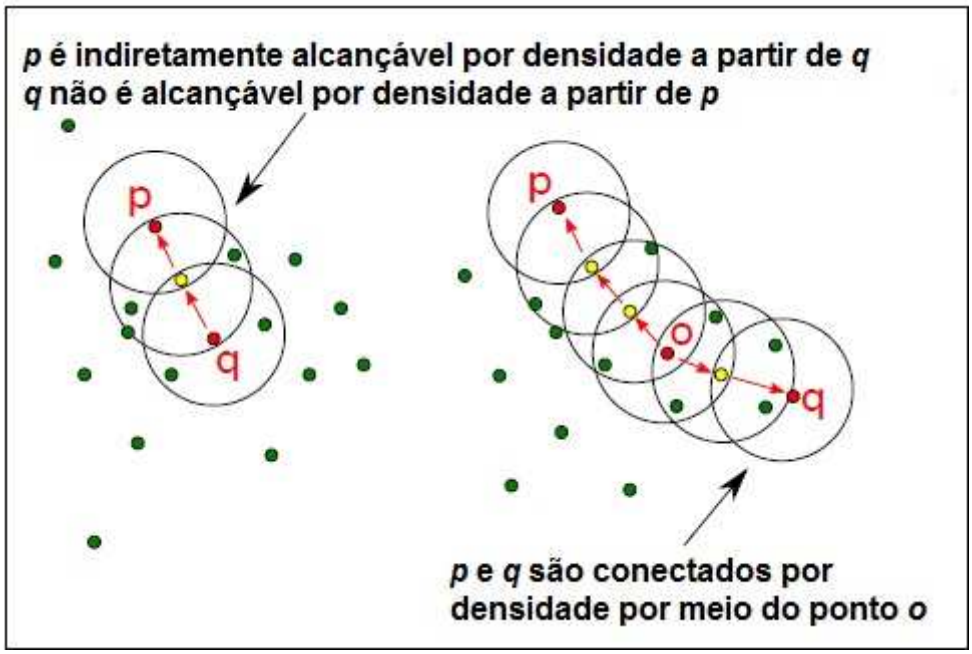

Figura 8. Pontos alcançáveis por densidade e pontos conectados por densidade
Fonte: Adaptado de ANKERST, M. et al (1999).

Os agrupamentos formados pelo algoritmo DBSCAN são baseados na idéia de que um *cluster* é um conjunto de todos os pontos conectados por densidade entre si, que representam o máximo com relação a pontos diretamente alcançáveis por densidade (ESTER et al, 1996, tradução nossa; KRIEGEL et al, 2011, tradução nossa). A noção de agrupamento baseado em densidade utilizada pelo algoritmo DBSCAN é apresentada a seguir.

4.5 *CLUSTERS* BASEADOS EM DENSIDADE E *OUTLIERS*

Dado um conjunto de pontos *D*, então um *cluster C* é um subconjunto não vazio de *D* que deve satisfazer as seguintes condições (ESTER et al, 1996, tradução nossa):

a) **maximalidade:** $\forall x_p, x_q \in D$: se $x_p \in C$ e x_q for indiretamente alcançável por densidade a partir de x_p então $x_q \in C$. Sendo *C* um *cluster* do conjunto de dados *D*, então cada ponto em *C* é diretamente alcançável por densidade a partir de qualquer um dos pontos centrais em *C*. Além disso, o

cluster C irá possuir todos os pontos que podem ser diretamente alcançáveis por densidade a partir de qualquer ponto central em *C*, obedecendo ao critério de maximalidade (ESTER et al, 1996, tradução nossa);

b) **conectividade:** $\forall x_p, x_q \in C$: *p* deve ser conectado por densidade a *q*, sendo que a relação de pontos conectados por densidade engloba todas as outras definições do algoritmo DBSCAN. Caso um *cluster* contiver apenas um ponto *p*, então esse ponto *p* deve estar conectado por densidade a si mesmo por meio de algum ponto x_i, que pode ser o próprio ponto *p*, obedecendo assim o critério de conectividade. Assim, o ponto x_i deve satisfazer a condição de ponto central, o que conseqüentemente implica que a $N_\varepsilon(x_i)$ possui ao menos η pontos (ESTER et al, 1996, tradução nossa).

Outliers são definidos como o conjunto de pontos em *D* que não estão inseridos em nenhum *cluster* (ESTER et al, 1996, tradução nossa).

Dados os parâmetros ε e η, inicialmente o algoritmo escolhe um ponto arbitrário x_p. Então toda a $N_\varepsilon(x_p)$ é recuperada e se x_p for um ponto de borda, não irão existir pontos diretamente alcançáveis por densidade a partir de x_p, pois a $N_\varepsilon(x_p) \leq \eta$. O ponto x_p é marcado como *outlier* e o DBSCAN visita o próximo ponto no conjunto de pontos.

Se um ponto for classificado como *outlier* pelo algoritmo, posteriormente ele pode estar na ε-vizinhança de outro ponto não visitado ainda pelo DBSCAN. Sendo assim, essa classificação pode ser removida caso o objeto seja diretamente alcançável por densidade a partir de um ponto central ainda não visitado.

Caso a $N_\varepsilon(x_p)$ contenha ao menos η pontos, um cluster *C* é criado contendo o ponto x_p e todos os pontos na ε-vizinhança de x_p. Formado o *cluster C*, a ε-vizinhança de cada ponto ainda não visitado em *C* é iterativamente recuperada e a densidade de cada ponto

nessa vizinhança é calculada, permitindo assim que novos pontos possam ser adicionados ao *cluster* (ANKERST et al, 1999, tradução nossa; ESTER et al, 1996, tradução nossa). A Figura 9 ilustra o processo de funcionamento do algoritmo de uma maneira geral:

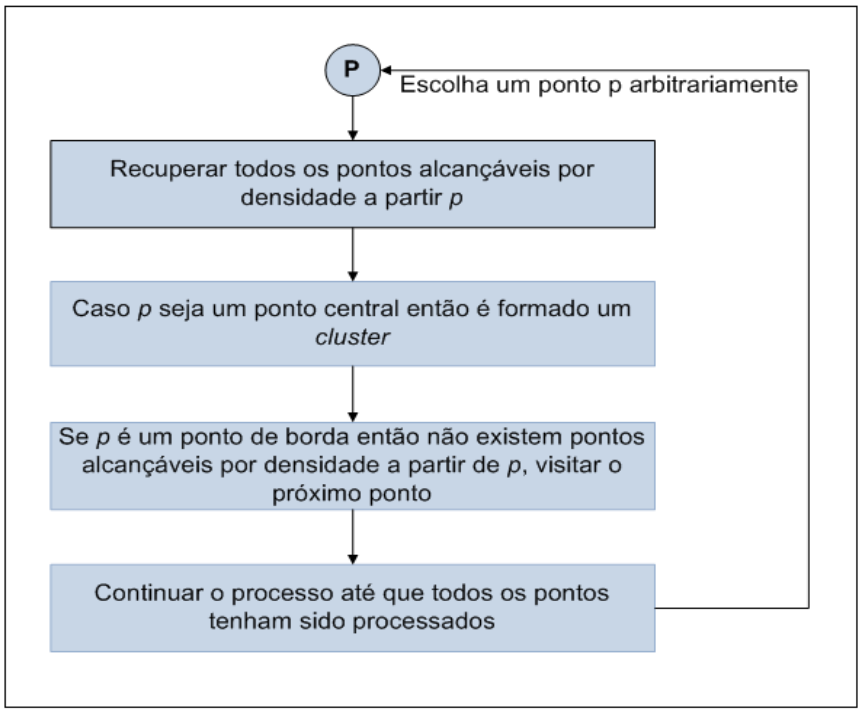

Figura 9. Funcionamento do algoritmo DBSCAN

Ao encontrar dois pontos centrais próximos, se a distância entre eles for menor que ε, ambos são colocados em um mesmo *cluster*. Os pontos de borda conseqüentemente serão colocados no mesmo *cluster* em que os pontos centrais estiverem (ESTER et al, 1996, tradução nossa; HAN; KAMBER, 2006, tradução nossa; OLIVEIRA, 2008).

A distância entre dois *clusters* C_1 e C_2 pode ser definida como a menor distância entre dois pontos x_p e x_q de tal maneira que o ponto x_p esteja contido em C_1 e x_q esteja contido em C_2 (ESTER et al, 1996, tradução nossa):

$$dist\ (C_1, C_2) = \min\{\ dist\ (x_p, x_q) \mid x_p \in C_1, x_q \in C_2\}$$

O funcionamento do DBSCAN pode ser melhor compreendido na Figura 10, onde *C1* e *C2* representam dois *clusters* encontrados pelo algoritmo DBSCAN. Considerando para o exemplo $\varepsilon = 3$, os pontos centrais são representados na figura pelos

pontos sólidos, e a ε-vizinhança de cada ponto é representada pelos círculos ao redor dos mesmos.

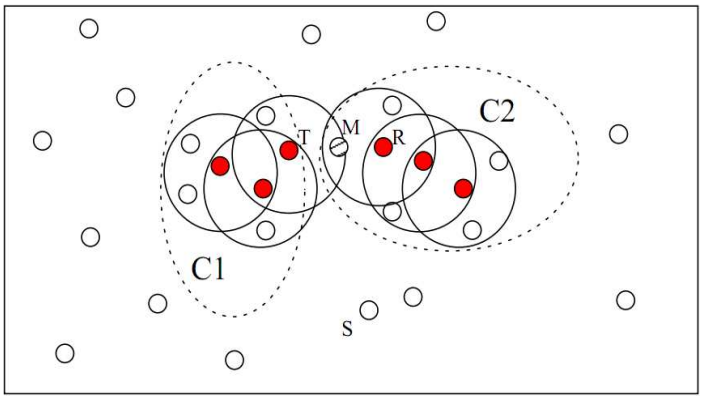

Figura 10. Dois *clusters* descobertos pelo DBSCAN
Fonte: HAN, J.; KAMBER, M.; TUNG, A. (2001)

Todos os objetos pertencentes a *C1* e *C2*, estão na ε-vizinhança de algum objeto central dos *clusters* a que pertencem e não existem dois objetos centrais de tal forma que um esteja na ε-vizinhança de outro que não pertencem ao mesmo *cluster*. Um objeto de borda como *M*, por exemplo, está na ε-vizinhança de dois objetos centrais, *T* e *R*, que pertencem respectivamente aos *clusters C1* e *C2*, podendo assim ser atribuído a qualquer um dos *clusters*, uma vez que está na região de fronteira de ambos os agrupamentos. Uma convenção utilizada para esses casos diz que o ponto *M* deverá ser atribuído ao *cluster* que foi primeiramente encontrado. O objeto *S* por sua vez é considerado *outlier*, pois não é um objeto central e não está na ε-vizinhança de nenhum objeto central (ESTER et al, 1996, tradução nossa; HAN; KAMBER; TUNG, 2001, tradução nossa).

O resultado do algoritmo DBSCAN é dependente da escolha adequada da medida de distância para o conjunto de dados, tendo em vista que o formato dos agrupamentos é determinado por essa função de distância (ESTER et al, 1996, tradução nossa; PATERLINI, 2011).

Essas medidas são usadas para calcular a densidade de cada ponto do conjunto de dados, ou seja, são contados quantos pontos estão contidos na ε-vizinhança de cada ponto

por meio de uma função de distância. Se a medida de distância entre um objeto x_p e um objeto x_q for menor que o parâmetro ε então o ponto x_q está contido na ε-vizinhança do objeto x_p. Considerando as diversas medidas de distância existentes, e as diversas formas de normalização de escalas de atributos, a seguir serão demonstradas as que foram implementadas nesse trabalho para o algoritmo DBSCAN.

4.6 FUNÇÕES DE DISTÂNCIA PARA A CONSULTA DE VIZINHANÇA

Para conjuntos de dados em que os atributos são contínuos[17] ou discretos[18], a distância euclidiana pode ser utilizada. Essa medida e adequada para conjuntos de dados que possuem grupos volumétricos aproximadamente esféricos (XU; WUNSCH, 2005, tradução nossa). Nesse tipo de medida, a distância entre dois objetos é denotada por:

$$d\left(x_{i,}x_j\right)=\sqrt{\sum_{l=1}^{d}\left(x_{il}-x_{jl}\right)^2}$$

Onde:

a) $d\left(x_{i,}x_j\right)$: é a distância euclidiana do objeto x_i para o objeto x_j ;

b) d: é o número de atributos presentes no conjunto de dados;

c) x_{il} : é o l-ésimo atributo do objeto x_i ;

d) x_{jl} : é o l-ésimo atributo do objeto x_j .

Diferentemente da distância euclidiana, à distância *Manhattan* troca as diferenças quadradas pela soma das diferenças absolutas entre os atributos e tende a formar agrupamentos com formato retangular (XU; WUNSCH, 2005, tradução nossa). Essa função

[17] Atributos contínuos podem assumir qualquer valor real dentro de um número pré-definido de valores (JAIN; DUBES, 1988, tradução nossa).

[18] Atributos discretos possuem, freqüentemente, um conjunto finito e pequeno de valores possíveis, como por exemplo, meses do ano (JAIN; DUBES, 1988, tradução nossa).

de distância corresponde ao somatório do módulo das diferenças entre os atributos e pode ser definida por:

$$d\left(x_{i,}x_{j}\right)=\sum_{l=1}^{d}\left|\left(x_{il}-x_{jl}\right)\right|$$

Onde:

a) $d\left(x_{i,}x_{j}\right)$: é à distância *Manhattan* do objeto x_i para o objeto x_j ;

b) *d*: é o número de atributos presentes no conjunto de dados;

c) x_{il} : é o l-ésimo atributo do objeto x_i ;

d) x_{jl} : é o l-ésimo atributo do objeto x_j .

Atributos diferentes podem ser medidos em escalas distintas. Portanto, caso for utilizada diretamente uma medida de distância como a euclidiana, por exemplo, atributos com escalas maiores irão se sobrepor a atributos medidos em escalas menores, tornando a clusterização tendenciosa, sendo que esse não é um problema especifico do algoritmo, mais das próprias medidas de distâncias (HAN; KAMBER, 2006, tradução nossa).

Considerando esse problema de escala, a normalização tem como objetivo colocar os valores para os atributos em um mesmo patamar. Isto faz com que medidas de distâncias utilizadas nos métodos de agrupamento possam estar na mesma escala, possuindo o mesmo peso, e assim evitando tendências na análise (HAN; KAMBER, 2006, tradução nossa; SHALABI; SHAABAN; KASASBEH, 2006, tradução nossa).

Portanto, antes de ser aplicada a função de distância, os atributos da base de dados são normalizados para o intervalo $[n\_\min, n\_\max]$, usando a normalização MIN-MAX. Essa função de normalização pode ser definida como (HAN; KAMBER, 2006, tradução nossa; SHALABI; SHAABAN; KASASBEH, 2006, tradução nossa; WITTEN; FRANK; HALL, 2011, tradução nossa):

$$z_i = \frac{v_i - \min_{vi}}{\max_{vi} - \min_{vi}} (n\_\max - n\_\min) + n\_\min$$

Onde:

a) z_i: representa o valor normalizado do *i*-ésimo atributo;

b) v_i: o valor do *i*-ésimo atributo;

c) $\min_{vi}$: menor valor encontrado para o *i*-ésimo atributo entre todos os registros da base de dados;

d) $\max_{vi}$: maior valor encontrado para o *i*-ésimo atributo entre todos os registros da base de dados;

e) $n\_\min$: novo menor intervalo para o atributo normalizado;

f) $n\_\max$: novo maior intervalo para o atributo normalizado.

Não existe uma regra geral que defina qual métrica de distância deve ser usada, nem qual o tipo de normalização que deve ser aplicado, sendo que essa escolha geralmente ocorre após a realização de vários testes com diferentes métricas. Uma ressalva pode ser feita caso se saiba antecipadamente o formato dos *clusters.* Assim, a distância mais apropriada será aquela que apresentar, para pontos eqüidistantes da origem, um formato semelhante aquele que é esperado para os agrupamentos (METZ, 2006).

Além da função de distância que será utilizada pelo algoritmo DBSCAN, existe a necessidade de ser informado o tamanho da região de vizinhança ε de um objeto e a número mínimo de objetos η que essa região de vizinhança deve conter, sendo que essa na maioria das vezes essa não é uma tarefa trivial.

Para a determinação do parâmetro ε é proposta em Ester et al (1996) uma heurística simples porém efetiva para a maioria dos casos para determinar os parâmetros globais de entrada do algoritmo.

4.7 DETERMINANDO OS PARÂMETROS DE ENTRADA PARA O DBSCAN

Sabendo que o algoritmo DBSCAN encontra agrupamentos baseado em parâmetros de densidade global, ou seja, utilizando-se dos mesmos valores ε e η para todos os *clusters*, pode-se deduzir que os valores desses parâmetros possuem impacto considerável nos resultados gerados (ESTER et al, 1996, tradução nossa).

Caso o valor para ε seja grande demais, então todos os pontos serão colocados em um único *cluster* e não serão detectados *outliers*. Por outro lado, caso seja um valor muito pequeno, todos os pontos serão classificados como *outliers* e nenhum agrupamento será encontrado. A Tabela 4 ilustra o efeito da escolha dos parâmetros de entrada sobre o resultado obtido.

Tabela 4. Parâmetros de entrada do DBSCAN

Valor para ε	**Valor para η**	**Resultado**
Alto	Alto	Poucos *clusters* grandes e densos
Baixo	Baixo	Muitos *clusters* pequenos e menos densos
Alto	Baixo	*Clusters* grandes e menos densos
Baixo	Alto	*Clusters* pequenos e densos

A fim de auxiliar o usuário na tarefa de escolha dos parâmetros de entrada para o algoritmo, então é proposto um método heurístico denominado função *k- dist*, que pode ser definida como (GAN; MA; WU, 2007, tradução nossa):

$$F_k\left(x_p\right)$$

Onde $F_k\left(x_p\right)$ representa a distância entre um ponto x_p é o seu *k*-ésimo vizinho mais próximo.

O primeiro passo do método heurístico consiste em calcular a função *k-dist* para todos os pontos do conjunto de dados *D*, tendo assim $F_k\left(D\right)$.

O passo seguinte consiste em ordenar decrescentemente $F_k\left(D\right)$, plotando-se

em um gráfico bidimensional que representa como ocorre a distribuição da densidade no conjunto de dados (ESTER et al, 1996, tradução nossa, GAN; MA; WU, 2007, tradução nossa).

Para os pontos que não estão em nenhum *cluster* são esperados valores relativamente altos para a função *k-dist*, enquanto que pontos localizados em algum grupo tendem a possuir valor baixo para essa função (Figura 11).

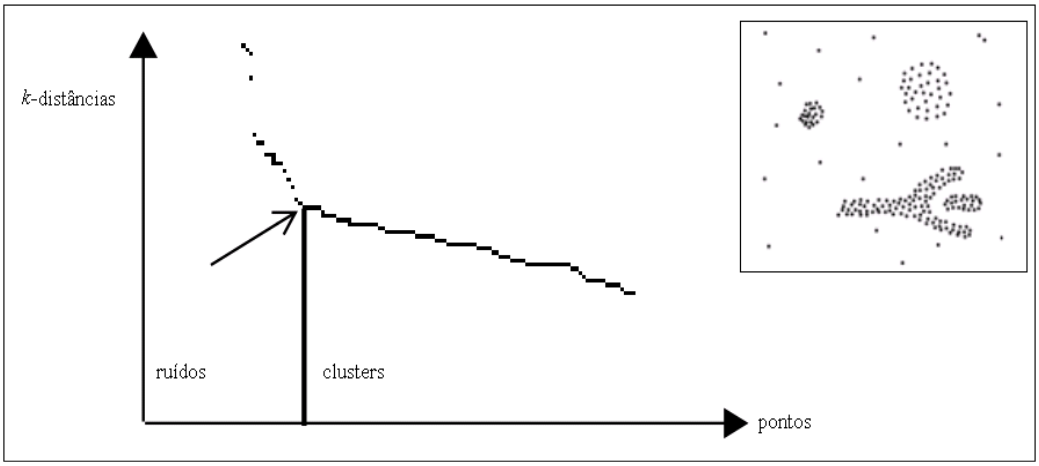

Figura 11. Gráfico representado as *k*-distâncias ordenadas de um conjunto de dados
Fonte: Adaptado de ESTER, M. et al (1996)

Em bases de dados bidimensionais, os autores propõem que seja usado o valor 4 para o parâmetro η, pois pesquisas indicam que valores acima disso não diferem significativamente nos resultados finais da clusterização (ESTER et al, 1996, tradução nossa; GAN; MA; WU, 2007, tradução nossa).

Para definição do parâmetro ε pode ser calculada a função *k-dist* para $k = \eta$, sendo $\eta = 4$, e usar para esse parâmetro o valor de $F_4\left(x_{p_i}\right)$, em que x_{p_i} é definido como o ponto inicial no primeiro "vale" do gráfico da função *k-dist*.

Tendo compreendido o funcionamento do algoritmo DBSCAN, no capitulo seguinte são apresentados alguns exemplos de aplicações desse algoritmo para a tarefa de clusterização em DM.

5 TRABALHOS CORRELATOS

O algoritmo DBSCAN tem sido bastante utilizado para a tarefa de clusterização, pois reage relativamente bem quando trabalha com conjuntos de dados contendo *outliers* e tem a capacidade de encontrar *clusters* com formatos variados, tornando se uma boa alternativa quando os dados possuem relacionamentos espaciais entre si.

Além disso, esse algoritmo encontra agrupamentos baseado nas propriedades dos dados, pois não requer que seja informado antecipadamente o número de *clusters* existentes no conjunto de dados. A seguir se encontram relacionados alguns exemplos de utilização do DBSCAN.

5.1 IDENTIFICAÇÃO DE REGIÕES COM CORES HOMOGÊNEAS EM IMAGENS BIOMÉDICAS

Este artigo foi desenvolvido por Emri Celebi, Alp Aslandogan e Paul Bergstresser no ano de 2005 sendo apresentado na *International Conference on Informaton Technology: Coding and Computing* (ITCC'05) em Washington nos Estados Unidos, e se propõe a aplicar o algoritmo DBSCAN na tarefa de identificação de regiões homogêneas de cores em imagens biomédicas, que representam tumores. O algoritmo DBSCAN foi escolhido porque possui a capacidade de encontrar *clusters* de formatos arbitrários enquanto preserva a proximidade espacial dos pontos de dados (CELEBI; ASLANDOGAN; BERGSTRESSER, 2005, tradução nossa).

Antes de serem submetidas ao algoritmo de clusterização, as imagens passaram por uma etapa de pré-processamento, pois imagens de lesão de pele geralmente contêm artefatos como a textura da pele e pêlos, tornando a segmentação mais complexa. Portanto,

com a finalidade de reduzir os efeitos provocados por esses artefatos, as imagens passam por um filtro de suavização, que tem a vantagem de preservar as bordas das lesões visíveis o suficiente, facilitando a sua detecção (CELEBI; ASLANDOGAN; BERGSTRESSER, 2005, tradução nossa).

As imagens também são subdividas em diversos conjuntos homogêneos e então são submetidas ao DBSCAN que iterativamente detecta as bordas das lesões e após isso, entra na região da lesão para identificar sub-regiões com cores diferentes, se as mesmas existirem (CELEBI; ASLANDOGAN; BERGSTRESSER, 2005, tradução nossa).

Figura 12. Regiões de lesão encontradas pelo DBSCAN
Fonte: CELEBI, E.; ASLANDOGAN, A.; BERGSTRESSER, P. (2005)

Os autores usaram um conjunto com 135 imagens de lesões de pele com dimensões de 256 x 256 para demonstrar a eficácia do método. Como forma de ajuste dos parâmetros de entrada do algoritmo, primeiramente foram selecionadas 18 imagens representativas do conjunto, e após isso, com os parâmetros selecionados de acordo com esse conjunto, o algoritmo foi aplicado nas 117 imagens restantes (CELEBI; ASLANDOGAN; BERGSTRESSER, 2005, tradução nossa).

A análise de resultados foi realizada por um especialista dermatologista e ficou constatado que em 80% dos casos, as lesões foram detectadas de maneira correta.

5.2 DETECÇÃO DE MICROCALCIFICAÇÕES EM MAMOGRAFIAS

Esse artigo foi desenvolvido no ano de 2005 por Anna Wróblewska, Arthur Przelaskowski, Pawel Bargiel e Piotr Boninski e publicado nos anais da *International Conference Of Medical Physics* (ICMP 2005) na cidade de Nuremberg, Alemanha, demonstrando um sistema de detecção e análise de agrupamentos de microcalcificações em exames de mamografia.

Nesse tipo de exame, as micro calcificações surgem como pequenas manchas ligeiramente mais brilhantes, em relação ao fundo variável do tecido circundante, sendo que calcificações malignas são pequenas (entre 0,05 milímetros e 1 milímitro), e possuem formato variável, enquanto calcificações benignas possuem tamanho maior e são mais suaves (WRÓBLEWSKA et al, 2005, tradução nossa).

Somente agrupamentos contendo mais que três partículas de calcificações em áreas de até 1cm² são consideradas suspeitas, sendo que quanto mais partículas existirem nessa área, maior a probabilidade de ser detectado o câncer (WRÓBLEWSKA et al, 2005, tradução nossa).

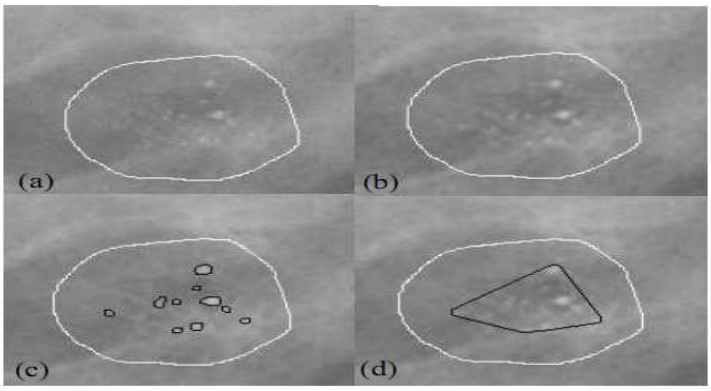

Figura 13. Microcalcificações detectadas pelo DBSCAN
Fonte: WRÓBLEWSKA, A. et al (2005)

No sistema proposto, o algoritmo DBSCAN é utilizado para realizar o agrupamento baseando-se na distribuição espacial e densidade dos objetos, pois suas

características garantem *clusters* com formatos mais precisos e, portanto, mais adequados para análise posterior por parte de especialistas (WRÓBLEWSKA et al, 2005, tradução nossa). Conforme os autores, em um grupo de 20 mamografias, o algoritmo identificou corretamente 83% dos casos.

5.3 AVALIAÇÃO DE TÉCNICAS DE AMOSTRAGEM EM GRANDES CONJUNTOS DE DADOS MULTIDIMENSIONAIS E CONTAMINADOS POR *OUTLIERS*

Na Tese de Doutorado de Ana Paula Appel concluída em 2010 pela Universidade de São Paulo (USP), na área de Ciência da Computação, o algoritmo DBSCAN é usado para avaliação de técnicas de pré-processamento aplicadas em grandes conjuntos de dados multidimensionais com características adversas, tais como *outliers* e densidades distintas na sua distribuição (APPEL, 2010).

Sabendo que as bases de dados possuem tamanho cada vez maior, a amostragem de dados é utilizada para viabilizar a complexidade da tarefa de detecção de agrupamentos, aumentando assim a velocidade dos algoritmos que realizarão essa tarefa. Porém, a maioria das técnicas de redução de dados estão baseadas na amostragem de dados uniforme, no qual cada elemento tem a mesma probabilidade de ser selecionado(APPEL, 2010).

Assim, uma nova técnica baseada na amostragem de dados balanceada pela densidade local denominada de *Biased Box Sampling* (BBS) foi proposta pela autora como alternativa na etapa de pré-processamento do KDD, mostrando-se eficiente na extração de amostragens balanceadas de conjuntos de dados com grandes variações no tamanho dos agrupamentos. Nessa amostragem, a probabilidade de um objeto ser incluído depende da densidade local do agrupamento (APPEL, 2010).

Na Figura 14 é possível visualizar em (a) um conjunto de dados contendo

aproximadamente 100.000 objetos distribuídos em cinco *clusters*, sendo que aproximadamente 10.000 desses pontos são *outliers*. Em (b) está uma amostra de 0,5% do conjunto original que foi obtida pelo algoritmo BBS, desenvolvido pela autora. Em (c) está exemplificado a amostra obtida pelo algoritmo *Density Biased Sampling*, em (d) os resultados do *Grid Biased Sampling* e por fim em (e), os resultados gerados pela técnica *Uniform Sampling*. O algoritmo BBS foi comparado com essas outras técnicas de amostragem com a finalidade de verificar a eficácia desse novo método, e como pode ser observado, a amostra gerada pelo BBS é a mais próxima do conjunto original de dados.

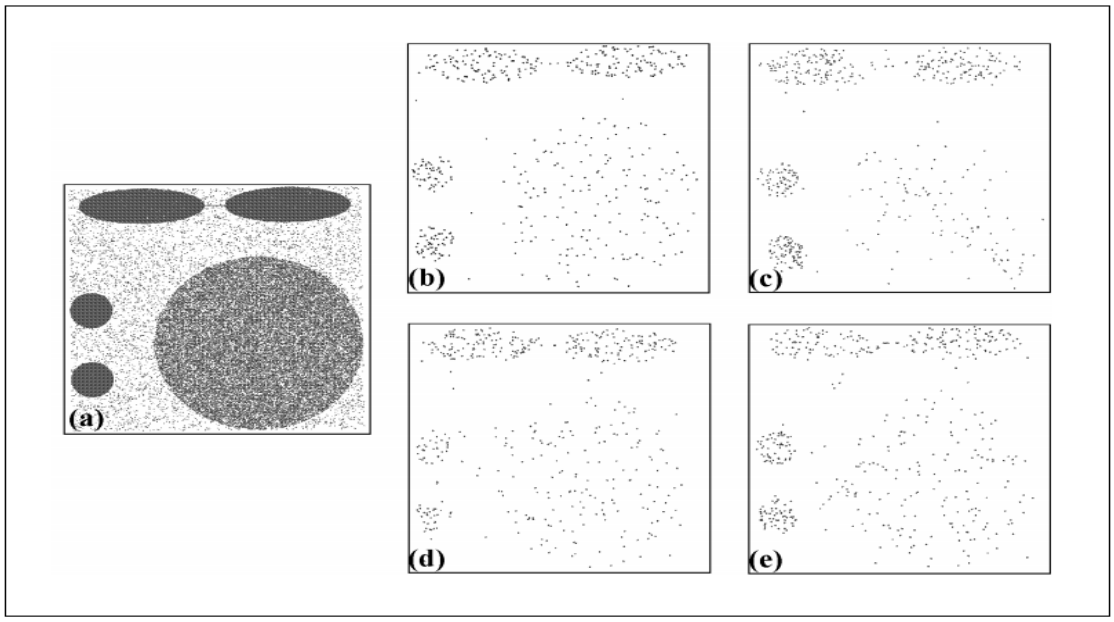

Figura 14. Agrupamentos encontrados pelo DBSCAN
Fonte: APPEL, A. (2010)

A avaliação dos resultados obtidos e a validação da técnica proposta foi realizada pelo DBSCAN, pois esse algoritmo não requer que o número de *clusters* seja pré-determinado como parâmetro, ou seja, ele pode encontrar agrupamentos baseado nas propriedades dos dados, o tornando apropriado para avaliar a qualidade das amostras obtidas pelo BBS. Outro fator importante na escolha do DBSCAN foi que ele é capaz de detectar *outliers*, o que não ocorre na maioria dos métodos hierárquicos e de particionamento (APPEL, 2010).

O ajuste do parâmetro η, necessário ao algoritmo, foi feito tendo como critério a taxa de amostragem aplicada ao número de elementos do menor *cluster* existente no conjunto de dados. Já para a determinação da ε-vizinhança o valor adotado foi o mesmo que o utilizado no conjunto de dados original (APPEL, 2010).

Tanto as amostras obtidas pela técnica desenvolvida pela autora quanto às obtidas pelas técnicas usadas para comparação com o algoritmo BSS foram submetidos ao DBSCAN, e o algoritmo conseguiu encontrou quatro *clusters* nas amostras, apesar do conjunto original possuir cinco grupos, pois, segundo a autora, existe uma ponte de *outliers* entre dois dos agrupamentos, o que os tornam conectados por densidade (APPEL, 2010).

5.4 CLUSTERIZAÇÃO DE TRÁFEGO DE REDE

No ano de 2006, Jeffrey Erman, Martin Arlitt e Anirban Mahanti publicaram nos anais da *SIGCOMM workshop on mining network data* (MineNet 2006) na cidade de Nova York nos Estados Unidos esse artigo, que apresenta um trabalho comparativo entre o algoritmo DBSCAN com outros dois algoritmos de clusterização, o *K-means* e o *AutoClass*, para a tarefa de clusterização de tráfego de rede.

A clusterização e a identificação precisa de tráfego de rede, de acordo com o tipo de protocolo utilizado é um importante elemento de muitas tarefas de gerenciamento de rede, tais como, priorização de fluxo, controle e policiamento de tráfego e geração de diagnósticos de monitoramento de rede (ERMAN; ARLITT; MAHANTI, 2006, tradução nossa).

Neste trabalho foram considerados os critérios de precisão na clusterização, ou seja, a capacidade que o algoritmo de clusterização possui de gerar *clusters* que contenham somente uma única categoria de protocolo e o tempo de processamento de cada algoritmo em particular. Os protocolos considerados no trabalho são: HTTP, P2P, POP3 e SMTP

(ERMAN; ARLITT; MAHANTI, 2006, tradução nossa).

Na comparação com os outros algoritmos de agrupamento, o DBSCAN se mostrou promissor, pois conseguiu separar a maior parte do tráfego de rede em um pequeno número de *clusters* de protocolos (ERMAN; ARLITT; MAHANTI, 2006, tradução nossa).

6 O ALGORITMO DBSCAN NA TAREFA DE CLUSTERIZAÇÃO DA *SHELL ORION DATA MINING ENGINE*

Mantida pelo Grupo de Pesquisa em Inteligência Computacional Aplicada, do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense, a *Shell Orion* é uma ferramenta de auxílio no processo de descoberta de conhecimento em bases de dados que tem constantemente suas funcionalidades aumentadas por meio da inserção de novos algoritmos e métodos por alunos em seus respectivos Trabalhos de Conclusão de Curso.

Dessa maneira, essa pesquisa se propõe a aumentar as funcionalidades presentes na *Shell Orion*, por meio do desenvolvimento do algoritmo DBSCAN para a tarefa de clusterização.

Para avaliação do desempenho e da qualidade dos algoritmos desenvolvidos na ferramenta, são usadas bases de dados, sendo que nessa pesquisa optou se pela utilização de uma base de dados da área ambiental, com informações sobre a qualidade das águas dos rios da região carbonífera catarinense.

6.1 BASE DE DADOS

A base de dados utilizada para avaliação do algoritmo DBSCAN apresenta indicadores ambientais da qualidade dos recursos hídricos das três bacias hidrográficas da região sul de Santa Catarina: Araranguá, Tubarão e Urussanga.

No sul do estado de Santa Catarina a indústria carbonífera tem historicamente grande importância, constituindo-se na base econômica de vários municípios. Por outro lado,

a mineração de carvão é reconhecida como uma das atividades com maior contribuição para a poluição ambiental da região, sobretudo dos seus recursos hídricos (PAVEI, 2007).

Dentre os processos associados à mineração de carvão o efluente resultante de reações de oxidação denominado de Drenagem Ácida de Mina (DAM), constitui-se em uma fonte causadora de severos impactos ao meio ambiente. Na região sul catarinense são encontrados cerca de 5.000 hectares de áreas degradadas pela atividade de mineração de carvão, estando 2/3 dos cursos da água da região comprometidos pela DAM (ALEXANDRE; KREBS; VIERO, 1995; GALATTO et al., 2007).

A DAM é proveniente de transformações ocorridas no rejeito da mineração de carvão, onde o sulfeto, oriundo de forma predominante da pirita[19], é inicialmente oxidado quimicamente e na seqüência do processo é catalisado por bactérias. A DAM é caracterizada por gerar efluentes com elevada acidez, baixo pH, e altas concentrações em metais dissolvidos, tais como, ferro, manganês, cobre e zinco, além de sulfatos (GALATTO et al., 2007; PAVEI, 2007).

Com o objetivo de promover a recuperação das áreas degradadas pela mineração e verificar o andamento das ações impostas a empresas do ramo, foi instituído o Grupo Técnico de Assessoramento (GTA) composto por representantes tanto das empresas mineradoras como da população e dos governos estadual e federal. Esse grupo divulga relatórios de monitoramento dos indicadores ambientais das bacias hidrográficas que tem por objetivo verificar a qualidade dos recursos hídricos da região carbonífera catarinense, a fim de avaliar a eficácia dos trabalhos de recuperação ambiental executados por empresas do ramo carbonífero (GTA, 2009).

A base de dados utilizada nesse trabalho faz referência ao terceiro relatório de monitoramento de indicadores ambientais, apresentado no ano de 2009 pelo GTA e possui

[19] Composto por enxofre e ferro (sulfeto de ferro), é o principal mineral capaz de produzir a DAM (GTA, 2009; PAVEI, 2007).

1313 registros sendo que cada registro possui 20 atributos que fazem referência a diversos indicadores de monitoramento dos recursos hídricos superficiais das três bacias hidrográficas da região sul catarinense. Este monitoramento é realizado de forma sistemática por meio de análises físico-química de água e de medidas de vazão em 140 pontos de monitoramento nas bacias hidrográficas dos rios Araranguá, Urussanga e Tubarão, distribuídos estrategicamente nas áreas impactadas pela mineração de carvão (GTA, 2009).

Na Figura 15 são mostradas as três bacias hidrográficas do sul catarinense com delimitação da região carbonífera, local onde os dados foram obtidos.

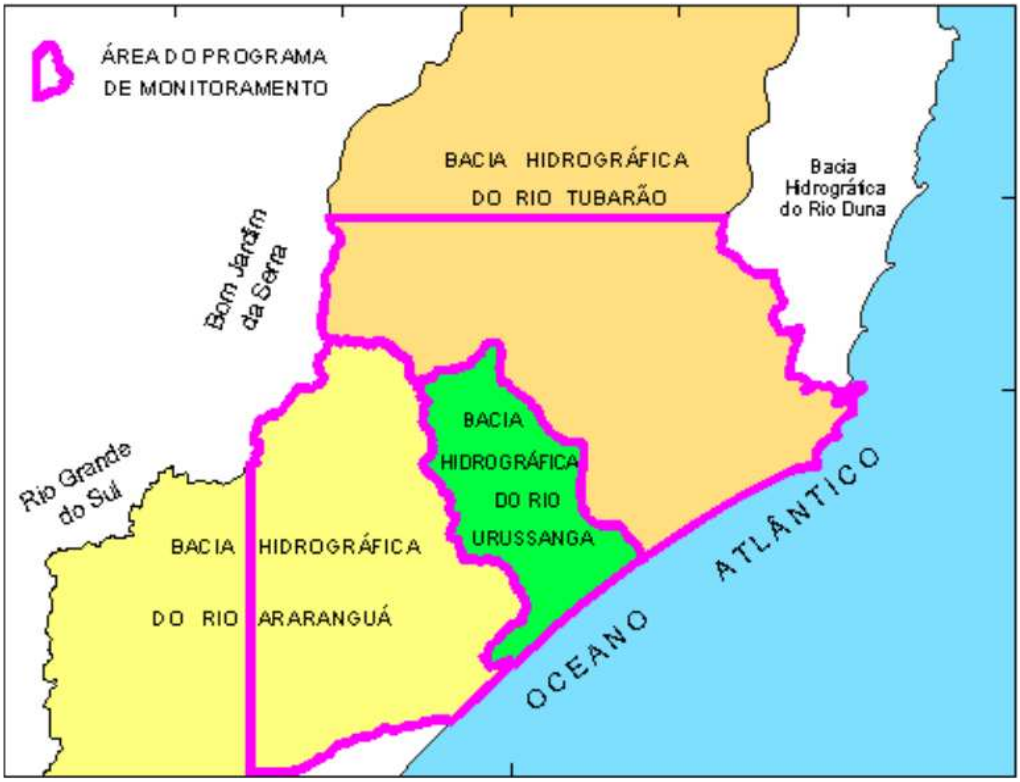

Figura 15. Bacias hidrográficas da região sul catarinense
Fonte: Adaptado de GTA (2009)

Dos 20 atributos da base de dados, 19 são numéricos e um diz respeito à data de coleta das informações, sendo que a mesma ocorreu entre março de 2002 a abril de 2009. A base de dados está normalizada, não existindo valores ausentes entre os atributos dos registros. Na Tabela 5 estão descritas as características de cada atributo, bem como as considerações técnicas que foram extraídas do relatório de monitoramento dos indicadores

ambientais (GTA, 2009).

Tabela 5. Base de dados de análise dos recursos hídricos

Atributo	Descrição	Valor
id	Atributo identificador do registro.	Número inteiro
id_cab	Identificador do ponto de monitoramento.	Número inteiro de 1 até 140
id_bacia	Atributo identificador da bacia hidrográfica onde o registro foi coletado.	1 para Araranguá, 2 para Tubarão e 3 para Urussanga
campanha	Identificador da campanha de monitoramento.	Número inteiro de 1 até 20
data	Atributo que faz referência a data de coleta do registro.	Data
ph	Grandeza físico-química conhecida como "potencial hidrogeniônico". É um valor entre 0 e 14 que indica se uma solução qualquer é ácida (*pH* < 7), neutra (*pH* = 7), ou alcalina (*pH* > 7). A faixa para o *pH* recomendável está entre 6 a 9.	Número decimal
vazão	Volume de água ou um fluido qualquer que passa, em uma determinada unidade de tempo, por meio de uma superfície.	Número decimal
acidez	Quantidade de ácido necessária para titular uma amostra a um determinado *pH*	Número decimal
cond	Condutividade elétrica da água, ou seja, capacidade da água conduzir corrente elétrica.	Número decimal
so, al, fe, mn	Respectivamente representam as concentrações de sódio, alumínio, ferro e manganês que estão presentes nas águas analisadas.	Número decimal
c_acidez	Carga de acidez. Calculada pela multiplicação da vazão pela concentração de acidez. Possui relação direta com a carga de contaminantes presentes na água.	Número decimal
c_fe, c_al, c_so, c_mn	Respectivamente representam as cargas de ferro, alumínio, sódio e manganês presentes nas águas.	Número decimal
precip	Precipitação regional (chuva) obtidos em estações meteorológicas das empresas carboníferas da região.	Número decimal
id_estacao	Atributo identificador da estação de monitoramento.	Número inteiro de 1 a 14

Fonte: Adaptado de GTA (2009).

Como os 1313 registros fazem referência às três bacias hidrográficas da região carbonífera catarinense, os registros estão subdivididos por bacia conforme a Tabela 6.

Tabela 6. Subdivisão da base dados por bacia hidrográfica

Bacia Hidrográfica	Total de registros	Pontos de coleta
Rio Araranguá	679	69
Rio Urussanga	425	37
Rio Tubarão	209	34

No que se refere à qualidade dos ecossistemas aquáticos, um dos complexos hídricos mais comprometidos pela atividade mineira na região carbonífera de Santa Catarina, é a Bacia hidrográfica do Rio Araranguá, onde grande parte do município de Criciúma está inserido, e onde está localizada cerca de 80% da produção de carvão da região (PAVEI, 2007). Por essas razões, nesse trabalho foram usados somente os dados de monitoramento coletados ao longo da bacia do rio Araranguá.

6.2 METODOLOGIA

As etapas metodológicas empregadas no desenvolvimento foram as seguintes: levantamento bibliográfico; modelagem por meio do padrão UML; demonstração matemática do algoritmo DBSCAN; implementação e realização de testes; análise de desempenho e validação dos resultados obtidos.

O levantamento bibliográfico levou em consideração a fundamentação e o entendimento de todos os temas envolvidos na pesquisa, tais como o processo de descoberta de conhecimento em bases de dados, *data mining*, o algoritmo DBSCAN e índices de avaliação de desempenho e qualidade para algoritmos de clusterização de dados.

6.2.1 Modelagem do Módulo do Algoritmo DBSCAN

O desenvolvimento teve inicio pela modelagem dos processos realizados pelo algoritmo DBSCAN por meio do padrão *Unified Modeling Language*[20] (UML). A modelagem UML proporciona uma melhor compreensão dos processos executados pelo algoritmo e dos passos necessários para sua execução. Para essa tarefa foi utilizada a

[20] Linguagem visual utilizada para modelar, especificar, visualizar, documentar e construir sistemas computacionais por meio do paradigma de Orientação a Objetos (FURLAN, 1998).

ferramenta *Astah Community*[21].

O diagrama de casos de uso permite que tenha uma idéia geral de como o sistema irá se comportar. Esse diagrama é o mais geral e informal da UML, sendo utilizado principalmente para auxiliar no levantamento e análise dos requisitos, em que são determinadas as necessidades do usuário e a compreensão do sistema como um todo (GUEDES, 2008):

a) **informar os parâmetros de entrada do algoritmo:** o usuário deve informar os parâmetros necessários para a execução do algoritmo. São informados o tamanho do raio da ε-vizinhança e o número mínimo de pontos nesse raio de vizinhança. O usuário também deve selecionar a função de distância desejada, sendo que estão disponíveis as distâncias euclidiana, euclidiana normalizada e *city-block*;
b) **execução do algoritmo:** a *Shell Orion* recebe os parâmetros de entrada selecionados pelo usuário e executa o DBSCAN retornando os resultados obtidos.

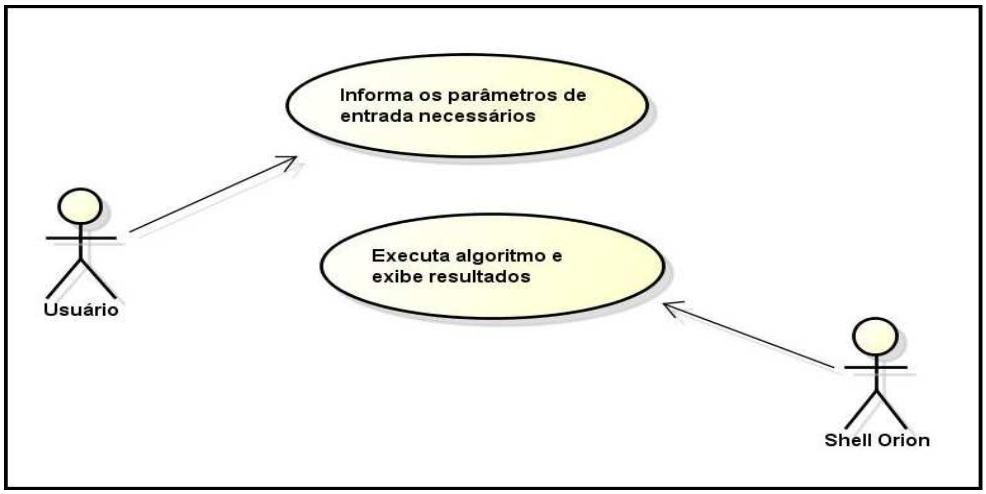

Figura 16. Diagrama de casos de uso

O diagrama de seqüência se preocupa com a ordem temporal em que as

[21] Disponível para *download* gratuitamente em (http://astah.net/download).

mensagens são trocadas entre os objetos envolvidos em um determinado processo. De modo geral, se baseia nos casos de uso para identificar os eventos geradores do processo modelado (GUEDES, 2008).

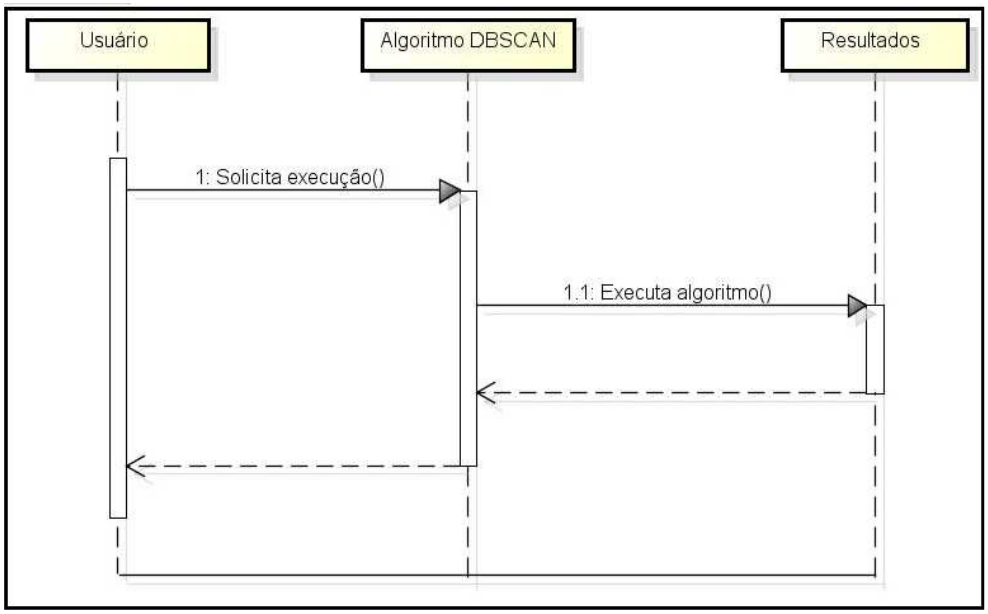

Figura 17. Diagrama de seqüência

Pode se observar por meio do diagrama de seqüência (Figura 17), a interação do usuário com o algoritmo DBSCAN. O usuário informa os parâmetros de entrada na tela inicial e solicita a execução do algoritmo, então o DBSCAN será executado e os resultados retornam para o usuário solicitante da operação.

Por fim, foi modelado o diagrama de atividades do algoritmo DBSCAN. Esse diagrama se preocupa em descrever os passos a serem percorridos para a conclusão de uma atividade específica (GUEDES, 2008).

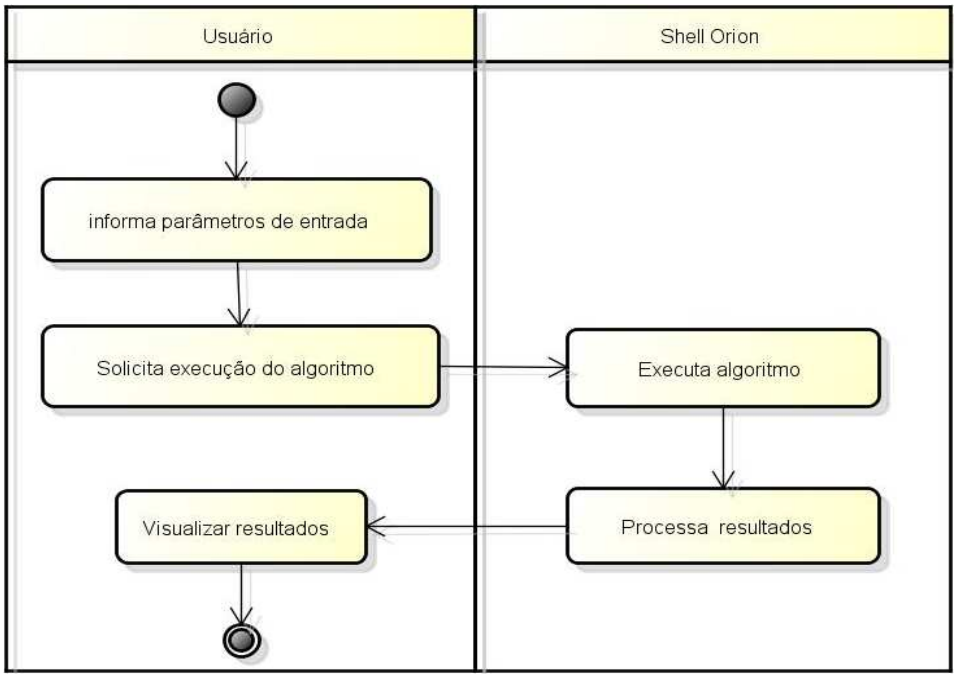

Figura 18. Diagrama de atividades

É possível verificar em um único processo (Figura 18) todo o fluxo de atividades necessárias para a execução do DBSCAN, sendo demonstrado como uma atividade depende da outra no fluxo de execução.

As atividades estão divididas entre as realizadas pelo usuário e as efetuadas pela *Shell Orion*. O usuário tem a tarefa de selecionar os parâmetros de entrada e solicitar a execução do algoritmo pela *Shell Orion*. Então a ferramenta executa o algoritmo e devolve os resultados obtidos para o usuário.

É importante salientar que a modelagem por meio da UML permitiu a melhor compreensão do sistema desenvolvido, bem como a visualização e documentação do funcionamento do algoritmo DBSCAN na *Shell Orion Data Mining Engine.*

6.2.2 Demonstração Matemática do Algoritmo DBSCAN

Nesta etapa do trabalho é demonstrado o algoritmo DBSCAN, por meio da modelagem matemática, permitindo a compreensão do seu funcionamento.

Os conceitos, técnicas e formalismos do algoritmo foram embasados no artigo escrito por Martin Ester, Peter Kriegel, Jörg Sander e Xiaowei Xu em 1996, intitulado *A Density-Based Algorithm for Discovering Cluster in Large Spacial Databases with Noise*.

Os valores fornecidos para os parâmetros de entrada do algoritmo e a escolha da medida de distância que será utilizada são de extrema importância, sendo que a saída resultante é consideravelmente afetada em função dessas escolhas.

Na demonstração do funcionamento do algoritmo foram definidos os seguintes parâmetros de entrada:

a) **raio da ε-vizinhança de um ponto:** especifica o valor para a ε-vizinhança de um objeto. Foi definido o valor 0,3 com a finalidade de simplificar a demonstração de funcionamento do algoritmo;

b) **número mínimo de pontos (η):** parâmetro que especifica o número mínimo de pontos, no dado raio de ε-vizinhança, que um ponto precisa possuir para ser considerado um ponto central. Definiu-se o valor 3 pois como a base de dados utilizada possui poucos registros, esse valor possibilita que seja encontrado mais de um *cluster*, demonstrando assim como ocorre o processo de criação de grupos pelo DBSCAN.

A função de distância escolhida para essa demonstração foi à euclidiana por ser de simples demonstração, assim possibilitando o melhor entendimento do funcionamento do algoritmo.

Nos Apêndices B e C tem-se a modelagem do algoritmo DBSCAN com as outras duas medidas de distância utilizadas nesse trabalho, respectivamente à distância *Manhattan* e a euclidiana normalizada.

Nesta demonstração matemática foi utilizada uma base de dados contendo 7 elementos (x_1, x_2, x_3, x_4, x_5, x_6, x_7) com 2 atributos cada um (a, b). A Tabela 7 exemplifica os

valores utilizados:

Tabela 7. Base de dados utilizada na modelagem do algoritmo

Atributos	x_1	x_2	x_3	x_4	x_5	x_6	x_7
a	5,1	4,9	4,7	4,6	5,0	5,3	8,0
b	3,5	3,0	3,2	3,1	3,6	3,5	3,0

Definida a base de dados a ser clusterizada, o primeiro passo do algoritmo consiste em montar uma estrutura denominada matriz de dissimilaridade. Em uma matriz de dissimilaridade é possível representar a distância entre pares de objetos. Essa matriz sempre será quadrada e de tamanho *n* x *n*, onde *n* representa a quantidade de objetos que serão clusterizados. A Figura 19 exemplifica uma matriz de dissimilaridade (HAN; KAMBER, 2006, tradução nossa):

$$\begin{bmatrix} 0 & & & & \\ dist\,(2,1) & 0 & & & \\ dist\,(3,1) & dist\,(3,2) & 0 & & \\ : & : & : & 0 & \\ dist\,(n,1) & dist\,(n,2) & .... & .... & 0 \end{bmatrix}$$

Figura 19. Matriz de dissimilaridade

Fonte: Adaptado de HAN, M.; KAMBER, J. (2006)

Conforme a Figura 19, *dist*(*i*,*j*) representa a distância ou dissimilaridade entre os objetos *i* e *j*. Geralmente *dist*(*i, j*) é um número não negativo que quanto mais próximo de zero for, mais similares serão os registros, caso contrário, quanto maior for esse valor, mais dissimilares serão os registros comparados. Pode-se destacar que a *dist*(*i j)* = *dist*(*j*,*i*) e que *dist*(*i*,*i*) = 0 (HAN; KAMBER, 2006, tradução nossa).

Essa matriz de dissimilaridade é descoberta empregando-se uma medida que é responsável pelo cálculo de distância para todos os pares de objetos da base de dados. Nesta demonstração empregou-se a distância euclidiana, como pode-se observar:

$$dist\ \left(x_{i,}\, x_{j}\right) = \sqrt{\sum_{l=1}^{d} \left(x_{il} - x_{jl}\right)^{2}}$$

$$dist\ (x_1, x_2) = \sqrt{(5,1 - 4,9)^2 + (3,5 - 3,0)^2} = \sqrt{0,04 + 0,25} = 0,5385$$
$$dist\ (x_1, x_3) = \sqrt{(5,1 - 4,7)^2 + (3,5 - 3,0)^2} = \sqrt{0,16 + 0,09} = 0,5000$$
$$dist\ (x_1, x_4) = \sqrt{(5,1 - 4,6)^2 + (3,5 - 3,1)^2} = \sqrt{0,25 + 0,16} = 0,6403$$
$$dist\ (x_1, x_5) = \sqrt{(5,1 - 5,0)^2 + (3,5 - 3,6)^2} = \sqrt{0,01 + 0,01} = 0,1414$$
$$dist\ (x_1, x_6) = \sqrt{(5,1 - 5,3)^2 + (3,5 - 3,5)^2} = \sqrt{0,04} = 0,2000$$
$$dist\ (x_1, x_7) = \sqrt{(5,1 - 8,0)^2 + (3,5 - 3,0)^2} = \sqrt{8,41 + 0,25} = 2,9428$$
$$dist\ (x_2, x_3) = \sqrt{(4,9 - 4,7)^2 + (3,0 - 3,2)^2} = \sqrt{0,04 + 0,04} = 0,2828$$
$$dist\ (x_2, x_4) = \sqrt{(4,9 - 4,6)^2 + (3,0 - 3,1)^2} = \sqrt{0,09 + 0,01} = 0,3162$$
$$dist\ (x_2, x_5) = \sqrt{(4,9 - 5,0)^2 + (3,0 - 3,6)^2} = \sqrt{0,01 + 0,36} = 0,6083$$
$$dist\ (x_2, x_6) = \sqrt{(4,9 - 5,3)^2 + (3,0 - 3,5)^2} = \sqrt{0,16 + 0,25} = 0,6403$$
$$dist\ (x_2, x_7) = \sqrt{(4,9 - 8,0)^2 + (3,0 - 3,0)^2} = \sqrt{9,61} = 3,1$$
$$dist\ (x_3, x_4) = \sqrt{(4,7 - 4,6)^2 + (3,2 - 3,1)^2} = \sqrt{0,01 + 0,01} = 0,1414$$
$$dist\ (x_3, x_5) = \sqrt{(4,7 - 5,0)^2 + (3,2 - 3,6)^2} = \sqrt{0,09 + 0,16} = 0,5000$$
$$dist\ (x_3, x_6) = \sqrt{(4,7 - 5,3)^2 + (3,2 - 3,5)^2} = \sqrt{0,36 + 0,09} = 0,6708$$
$$dist\ (x_3, x_7) = \sqrt{(4,7 - 8,0)^2 + (3,2 - 3,0)^2} = \sqrt{10,89 + 0,04} = 3,3061$$
$$dist\ (x_4, x_5) = \sqrt{(4,6 - 5,0)^2 + (3,1 - 3,6)^2} = \sqrt{0,16 + 0,25} = 0,6403$$
$$dist\ (x_4, x_6) = \sqrt{(4,6 - 5,3)^2 + (3,1 - 3,5)^2} = \sqrt{0,49 + 0,16} = 0,8062$$
$$dist\ (x_4, x_7) = \sqrt{(4,6 - 8,0)^2 + (3,1 - 3,0)^2} = \sqrt{11,56 + 0,01} = 3,4015$$
$$dist\ (x_5, x_6) = \sqrt{(5,0 - 5,3)^2 + (3,6 - 3,5)^2} = \sqrt{0,09 + 0,01} = 0,3162$$
$$dist\ (x_5, x_7) = \sqrt{(5,0 - 8,0)^2 + (3,6 - 3,0)^2} = \sqrt{9,0 + 0,36} = 3,0594$$
$$dist\ (x_6, x_7) = \sqrt{(5,3 - 8,0)^2 + (3,5 - 3,0)^2} = \sqrt{7,29 + 0,25} = 2,7459$$

Calculada a distância entre todos os pares de objetos da base de dados, então o algoritmo monta a matriz de dissimilaridade:

$$\begin{bmatrix} 0 & 0,5385 & 0,5000 & 0,6403 & 0,1414 & 0,2000 & 2,9428 \\ 0,5385 & 0 & 0,2828 & 0,3162 & 0,6083 & 0,6403 & 3,1000 \\ 0,5000 & 0,2828 & 0 & 0,1414 & 0,5000 & 0,6708 & 3,3061 \\ 0,6403 & 0,3162 & 0,1414 & 0 & 0,6403 & 0,8062 & 3,4015 \\ 0,1414 & 0,6083 & 0,5000 & 0,6403 & 0 & 0,3162 & 3,0594 \\ 0,2000 & 0,6403 & 0,6708 & 0,8062 & 0,3162 & 0 & 2,7459 \\ 2,9428 & 3,1000 & 3,3061 & 3,4015 & 3,0594 & 2,7459 & 0 \end{bmatrix}$$

Tendo a matriz de dissimilaridade montada, o próximo passo executado pelo algoritmo consiste em verificar a ε-vizinhança de cada ponto da base de dados com a finalidade de identificar possíveis pontos centrais para iniciar a formação dos agrupamentos.

Um conjunto denominado C_a contendo todos os pontos da base de dados é montado. Nesse conjunto os objetos recebem a marcação de não classificados. Então o algoritmo irá visitar cada ponto nesse conjunto e terminará quando não mais existirem elementos em C_a.

Esse conjunto auxiliar C_a fica da seguinte maneira:

$$C_a = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$$

O algoritmo escolhe o primeiro ponto no conjunto C_a, ou seja, x_1. Então a $N_\varepsilon(x_1)$ para saber se a cardinalidade do ponto é igual ou excede η, conferindo a condição: $Card\ (N_\varepsilon(x_1)) \geq \eta$. Para isso, o algoritmo DBSCAN consulta a matriz de dissimilaridade a fim de saber quantos pontos a ε-vizinhança de x_1 contêm:

$$dist\,(x_1, x_2) = 0{,}5385$$
$$dist\,(x_1, x_3) = 0{,}5000$$
$$dist\,(x_1, x_4) = 0{,}6403$$
$$dist\,(x_1, x_5) = 0{,}1414$$
$$dist\,(x_1, x_6) = 0{,}2000$$
$$dist\,(x_1, x_7) = 2{,}9428$$

Consultando a $N_\varepsilon(x_1)$ o DBSCAN constatou que os pontos x_5 e x_6 estão na ε-vizinhança de x_1 visto que $dist(x_1, x_5) \leq \varepsilon$ e $dist(x_1, x_6) \leq \varepsilon$ para ε= 0.3 e, portanto são diretamente alcançáveis por densidade a partir de x_1. Isso implica que x_1 é um ponto central, pois a $N_\varepsilon(x_1) \geq \eta$ para η = 3.

Assim, um *cluster* C_1 é formado contendo o ponto central x_1 e todos os pontos diretamente alcançáveis por densidade a partir de x_1, ou seja, todos os pontos na $N_\varepsilon(x_1)$.

O *cluster* C_1 é formado inicialmente pelos seguintes pontos:

$$C_1 = \{x_1, x_5, x_6\}$$

O algoritmo irá remover o ponto x_1 do conjunto C_a que agora irá ficar da

seguinte maneira:

$$C_a = \{x_2, x_3, x_4, x_5, x_6, x_7\}$$

O próximo passo consiste em recuperar a ε-vizinhança dos pontos no *cluster* C_1 que ainda não foram visitados. Iniciando por x_5, o algoritmo consulta a matriz de dissimilaridade a fim de obter a condição desse ponto:

$$dist(x_5, x_1) = 0{,}1414$$
$$dist(x_5, x_2) = 0{,}6083$$
$$dist(x_5, x_3) = 0{,}5000$$
$$dist(x_5, x_4) = 0{,}6403$$
$$dist(x_5, x_6) = 0{,}3162$$
$$dist(x_5, x_7) = 3{,}0594$$

Verificou-se que a $N_\varepsilon(x_5)$ não possui ao menos η, pois somente x_1 se encontra na ε-vizinhança de x_5 para $\varepsilon = 0.3$.

Visto que x_5 é um ponto de borda em C_1, não existem pontos novos para serem adicionados ao *cluster* e o DBSCAN atualiza o conjunto C_a removendo o ponto x_5:

$$C_a = \{x_2, x_3, x_4, x_6, x_7\}$$

Caso x_5 fosse um ponto central e existissem pontos na $N_\varepsilon(x_5)$ que ainda não estivessem inseridos em C_1, o algoritmo iria adicionar esses pontos a C_1 e continuar recursivamente o processo de verificação de todos os pontos contidos em C_1 até que não existisse mais nenhum ponto em C_1 que ainda não tivesse sido visitado.

Assim, o DBSCAN visita o próximo ponto não visitado em C_1, ou seja, x_6.

Verificando a $N_\varepsilon(x_6)$ tem-se:

$$dist(x_6, x_1) = 0{,}2000$$
$$dist(x_6, x_2) = 0{,}6403$$
$$dist(x_6, x_3) = 0{,}6708$$
$$dist(x_6, x_4) = 0{,}8062$$
$$dist(x_6, x_5) = 0{,}3162$$
$$dist(x_6, x_7) = 2{,}7459$$

Como aconteceu no ponto x_5, nenhum novo ponto foi encontrado para ser adicionado a C_1, tendo em vista que x_1 está na $N_\varepsilon(x_6)$, porem não é diretamente alcançável por densidade a partir de x_6, pois o ponto x_6 não é um ponto central no *cluster* C_1, visto que em sua ε-vizinhança não estão contidos ao menos η pontos.

Atualizando o conjunto auxiliar:

$$C_a = \{x_2, x_3, x_4, x_7\}$$

Foram visitados todos os pontos desse primeiro *cluster* formado e nenhum novo ponto foi adicionado. Portanto, um primeiro *cluster* é constituído da seguinte maneira:

$$C_1 = \{x_1, x_5, x_6\}$$

Algumas propriedades podem ser identificadas no *cluster* C_1:

a) o ponto x_1 é um ponto central em C_1, pois sua ε-vizinhança possui ao menos um número η de pontos (x_5 e x_6);
b) os pontos x_5 e x_6 são definidos como pontos de borda, devido ao fato de que a esses dois pontos não possuem em sua ε-vizinhança o número mínimo de pontos η necessários para serem considerados pontos centrais;
c) o ponto x_5 e o ponto x_6 são diretamente alcançáveis por densidade a partir de x_1, pois x_1 é um ponto central;
d) o ponto x_1 não é diretamente alcançável por densidade a partir de x_5 e nem a partir de x_6, pelo fato de que x_5 e x_6 não são pontos centrais em C_1;
e) o ponto x_5 não é diretamente alcançável por densidade a partir de x_6, visto que x_6 não é um ponto central;
f) o ponto x_5 é conectado por densidade ao ponto x_6, pois de acordo com a definição de *cluster* baseado em densidade, existe em C_1 um ponto (x_1) a partir do qual ambos os pontos são alcançáveis por densidade, tendo-se

portanto uma relação simétrica.

A partir das propriedades identificadas em C_1 fica mais clara a noção de *cluster* baseado em densidade, sendo que um *cluster* é definido como um conjunto de pontos conectados por densidade.

Formado o cluster C_1, o DBSCAN recupera a ε-vizinhança do próximo ponto da base de dados. Verificando o conjunto C_a, o próximo ponto a ser visitado é x_2.

É importante notar que o algoritmo não encontra mais a necessidade de verificar a distância de pontos que já estão incluídos em C_1, sabendo que não existe mais possibilidade de inclusão de novos pontos em *clusters* já terminados. Assim, os pontos x_1, x_5 e x_6 não precisam ser verificados.

Recuperando a $N_\varepsilon(x_2)$:

$$dist\,(x_2, x_3) = 0{,}2828$$
$$dist\,(x_2, x_4) = 0{,}3162$$
$$dist\,(x_2, x_7) = 3{,}1000$$

Foi constatado que x_2 não possui ao menos η pontos na sua ε-vizinhança. Somente x_3 está na ε-vizinhança de x_2 e assim a condição $N_\varepsilon(x_2) \geq \eta$ não é satisfeita.

Desse modo, o DBSCAN marca momentaneamente o ponto x_2 como *outlier* e atualiza o conjunto C_a:

$$C_a = \{x_3, x_4, x_7\}$$

O algoritmo DBSCAN irá considerar como *outlier* qualquer ponto na base de dados que não pertença a nenhum *cluster*. No caso do ponto x_2, o algoritmo o considerou como *outlier* momentaneamente, pois x_2 não é um ponto central e não está contido em nenhum *cluster*. Porém, se após visitar outro ponto p qualquer na base de dados e for verificado que x_2 está na ε-vizinhança desse ponto p e a $N_\varepsilon(x_p) \geq \eta$ então a marcação de *outlier* será removida e x_2 será adicionado ao *cluster* que contêm a ε-vizinhança de p.

Visitando o próximo ponto não visitado da base de dados, é recuperada a $N_{\varepsilon}(x_3)$:

$$dist\,(x_3, x_2) = 0{,}2828$$
$$dist\,(x_3, x_4) = 0{,}1414$$
$$dist\,(x_3, x_7) = 3{,}3061$$

Constatou se que x_3 é um ponto central, pois a $N_{\varepsilon}(x_3) \geq \eta$. Além de x_3 estar conectado a si mesmo obedecendo a propriedade de conectividade, x_2 e x_4 são diretamente alcançáveis por densidade a partir de x_3.

Um novo *cluster* C_2 é criado tendo como ponto central x_3, além dos pontos x_2 e x_4:

$$C_2 = \{x_2, x_3, x_4\}$$

O ponto x_2 que anteriormente foi classificado como *outlier* agora é atribuído ao *cluster* C_2 visto que está contido na ε-vizinhança de um ponto central (x_3).

O ponto x_3 é removido do conjunto de pontos não visitados, tendo-se como conjunto auxiliar:

$$C_a = \{x_4, x_7\}$$

O DBSCAN irá assim visitar o próximo ponto não visitado em C_2, que é o ponto x_4:

$$dist\,(x_4, x_2) = 0{,}3162$$
$$dist\,(x_4, x_3) = 0{,}1414$$
$$dist\,(x_4, x_7) = 3{,}4015$$

O ponto x_4 é classificado como ponto de borda em C_2 visto que não possui o número mínimo de pontos em sua ε-vizinhança. Portanto, não existem mais pontos a serem visitados em C_2 e mais um *cluster* é terminado contendo x_3 como ponto central e os pontos x_2 e x_4 como pontos de borda, sendo diretamente alcançáveis por densidade a partir de x_3.

$$C_2 = \{x_2, x_3, x_4\}$$

Atualizando o conjunto auxiliar:

$$C_a = \{x_7\}$$

O DBSCAN nesse momento constata que o conjunto C_a possui somente o ponto x_7 e não existe mais condição de x_7 formar um *cluster*, visto que o número mínimo de pontos necessários para a formação de um agrupamento foi definido como 3, e nem modo de x_7 se inserir em um grupo já formado, uma vez que se esse ponto estivesse na ε-vizinhança de algum ponto central de um dos *clusters*, ele também estaria inserido nesse *cluster*, obedecendo o critério de maximalidade de cluster baseado em densidade. Portanto, como x_7 possui atributos com valores muito divergentes dos demais pontos da base de dados, ele não foi atribuído a nenhum agrupamento e o algoritmo o classifica como um *outlier*, atualizando o conjunto C_a:

$$C_a = \{\emptyset\}$$

Verificando o conjunto C_a o DBSCAN não encontra mais nenhum ponto a ser visitado na base de dados e encerra a sua execução nesse momento.

Foram encontrados dois *clusters* e um ponto foi classificado como *outlier*, como segue abaixo:

$$C_1 = \{x_1, x_5, x_6\}$$
$$C_2 = \{x_2, x_3, x_4\}$$
$$Outlier = \{x_7\}$$

Importante notar que caso fosse escolhida outra medida de distância para montar a matriz de dissimilaridade, os resultados tenderiam a ser consideravelmente diferentes dos encontrados quando aplicada a distância euclidiana.

Outro fator que pode alterar consideravelmente o resultado obtido são os valores para os parâmetros de entrada. Nessa demonstração, caso fosse informado para o parâmetro

η o valor 4, o algoritmo iria classificar todos os pontos como *outlier*, visto que não existe nem um ponto dessa base de dados que possui 4 vizinhos em sua ε-vizinhança para ε=0,3. Portanto os valores para os parâmetros de entrada merecem atenção especial.

A compreensão dos cálculos e do fluxo de funcionamento do algoritmo possibilitou a sua implementação na *Shell Orion Data Mining Engine.*

6.2.3 Índices Empregados na Validação

A clusterização de dados é um processo não supervisionado, em que não existem grupos pré-definidas e exemplos que possam mostrar se os *clusters* encontrados por um determinado algoritmo são significativos (GAN; MA; WU, 2007, tradução nossa).

Como forma de auxílio na determinação da qualidade dos grupos encontrados por algoritmos de clusterização, o resultado do agrupamento deve ser validado com o objetivo de verificar a solução encontrada, sendo que essa tarefa é feita geralmente por meio de índices estatísticos (JAIN; DUBES, 1988, tradução nossa).

Após pesquisa na literatura, com a finalidade de verificar quais os índices de validação que poderiam ser usados para avaliar a qualidade das partições geradas pelo DBSCAN, foram definidos alguns índices de validação a serem empregados, como: índice de *Dunn* e o *C-Index*.

O índice de *Dunn* foi utilizado porque permite à avaliação dos *clusters* com relação à compactação interna e separação externa, identificando o quanto os grupos encontrados são densos e separados dos outros (BEZDEK, 2005, tradução nossa; JAIN; DUBES, 1988, tradução nossa). Já o *C-Index* foi escolhido por permitir que a homogeneidade de cada *cluster* encontrado seja avaliada separadamente, ou seja, com esse índice é possível verificar até que ponto, objetos similares foram colocados em um mesmo

cluster (MILLIGAN; COOPER, 1985, tradução nossa).

6.2.3.1 Índice de *Dunn*

Baseando-se na idéia que um *cluster* consiste em uma coleção de objetos que são similares entre si e dissimilares com os objetos de outros *clusters*, o índice de *Dunn* foi proposto por J.C. Dunn em 1974 e avalia as partições geradas com o objetivo de identificar o quanto os *clusters* encontrados são compactos e bem separados (LAROSE, 2005, tradução nossa; JAIN; DUBES, 1988, tradução nossa).

Caso o conjunto de dados contenha grupos compactos e bem separadas, é esperado que a diferença entre os *clusters* seja grande e o diâmetro dos grupos seja pequeno. Sendo assim, e baseando se nas definições do índice de *Dunn*, pode-se dizer que valores altos para o índice indicam a presença de *clusters* compactos e bem separados no conjunto de dados (JAIN; DUBES, 1988, tradução nossa).

O índice de *Dunn* pode ser definido como o valor da razão da menor distância entre dois *clusters* distintos, pelo maior valor encontrado para o diâmetro de um *cluster* presente no conjunto dos dados. O valor do diâmetro indica a dispersão interna de um agrupamento:

$$Dunn = \min_{1<i<k} \left(\min_{i+1 \le j \le k} \left(\frac{dist\left(c_i, c_j\right)}{\max\limits_{1 \le l \le k} diam\left(c_l\right)} \right) \right)$$

Onde:

a) *Dunn*: índice de *Dunn*;

b) $dist(c_i, c_j)$**:** distância entre o *i*-ésimo e o *j*-ésimo *cluster* de tal forma que $dist(c_i, c_j) = \min_{x_i \in c_i, x_j \in c_j} dist(x_i, x_j)$;

c) *k*: número total de *clusters*;

d) $diam(c_l)$**:** dispersão do *l*-ésimo *cluster* onde $diam(c_l) = \max_{x_{l_1}, x_{l_2} \in c_l} dist(x_{l_1}, x_{l_2})$.

É importante salientar que o índice de *Dunn* é bastante sensível a presença de *outliers* entre os dados e por essa razão pontos assim classificados pelo DBSCAN não são considerados no cálculo. Também devem ser encontrados no mínimo dois *clusters* para que este índice seja empregado.

A demonstração desse índice é feita usando os resultados encontrados na modelagem matemática do algoritmo. Portanto, considerando-se os *clusters* encontrados pelo DBSCAN:

$$C_1 = \{x_1, x_5, x_6\}$$
$$C_2 = \{x_2, x_3, x_4\}$$

Calculando-se a distância entre os *clusters* C_1 e C_2:

$$dist(x_1, x_2) = 0{,}5385$$
$$dist(x_1, x_3) = 0{,}5000$$
$$dist(x_1, x_4) = 0{,}6403$$
$$dist(x_5, x_2) = 0{,}6083$$
$$dist(x_5, x_3) = 0{,}5000$$
$$dist(x_5, x_4) = 0{,}6403$$
$$dist(x_6, x_2) = 0{,}6403$$
$$dist(x_6, x_3) = 0{,}6708$$
$$dist(x_6, x_4) = 0{,}8062$$

Portanto tem-se $\min_{x_i \in c_1, x_j \in c_2} dist(x_i, x_j) = 0{,}5000$, sendo essa a distância calculada pelo índice entre os *clusters* C_1 e C_2. Caso fossem encontrados mais que dois *clusters*, a distância entre todos os pares de pontos desses agrupamentos seria calculada e a menor distância seria selecionada pelo índice para representar a separação externa dos *clusters*.

O passo seguinte consiste em encontrar o *cluster* com maior diâmetro, ou seja, cada grupo encontrado tem sua dispersão interna verificada, e a maior distância entre dois pontos no mesmo *cluster* é selecionada como sendo a dispersão desse grupo. Iniciando com o *cluster* C_1:

$$dist\ (x_1, x_5) = 0{,}1414$$
$$dist\ (x_1, x_6) = 0{,}2000$$
$$dist\ (x_6, x_5) = 0{,}3162$$

Calculando a dispersão do *cluster* C_2:

$$dist\ (x_2, x_3) = 0{,}2828$$
$$dist\ (x_2, x_4) = 0{,}3162$$
$$dist\ (x_4, x_3) = 0{,}1414$$

Tem se a seguinte dispersão para cada um dos *clusters*:

$$diam\,(c_1) = 0{,}3162$$
$$diam\,(c_2) = 0{,}3162$$

Constatou-se que os dois *clusters* possuem o mesmo valor para a dispersão interna, sendo que caso esses valores fossem diferentes, seria utilizado como diâmetro pelo índice de *Dunn* o maior valor para a dispersão de um grupo. Portanto tem-se o seguinte valor para o diâmetro do índice de *Dunn*:

$$\max_{x_{l_1}, x_{l_2} \in c_l} dist\left(x_{l_1}, x_{l_2}\right) = 0{,}3162$$

Resolvendo-se a equação do índice de *Dunn*, encontra-se como valor para esse exemplo:

$$Dunn = \left(\frac{0{,}5000}{0{,}3162}\right)$$
$$Dunn = 1.581277672359266$$

Sabendo que esse índice deve ser maximizado, pode se concluir que o resultado encontrado indicou que os *clusters* encontrados pelo DBSCAN na modelagem matemática são compactos e estão bem separados uns dos outros.

6.2.3.2 *C-Index*

O *C-Index* foi proposto por L. J. Hubert e J. R. Levin em 1976, e ao contrário do índice de *Dunn*, apresenta valores no intervalo $[0,1]$, é calculado para cada *cluster* individualmente e valores baixos indicam uma boa clusterização dos dados (MILLIGAN; COOPER, 1985, tradução nossa).

Nesse índice a coesão interna de um *cluster* é verificada, sendo que quanto mais semelhantes forem os pontos contidos em um agrupamento, ou seja, quanto mais compacto for o *cluster*, mais próximo de zero será o resultado do *C-Index* para o grupo (MILLIGAN; COOPER, 1985, tradução nossa).

Tendo *n* como o número de pares de pontos presentes em um determinado *cluster*, então o *C-Index* é definido como sendo o valor da razão da soma das distâncias entre todos os pares de pontos presentes nesse agrupamento menos a soma das *n* menores distâncias entre todos os objetos da base de dados pela soma das *n* maiores distâncias entre todos os pares de pontos da base de dados menos a soma das *n* menores distâncias entre todos os pares de objetos da base:

$$C - Index = \frac{d_w(C_i) - \min(n_w)}{\max(n_w) - \min(n_w)}$$

Onde:

a) $d_w(C_i)$: somatório das distâncias entre todos os pares de pontos no *cluster* C_i;

b) n_w : quantidade de pares de pontos no *cluster* C_i ;

c) $\min(n_w)$: somatório das n_w menores distâncias entre os pares de objetos do conjunto de dados;

d) $\max(n_w)$: somatório das n_w maiores distâncias entre os pares de objetos do conjunto de dados.

A soma das distâncias entre todos os pares de pontos no *cluster* C_i é definida por:

$$d_w(C_i) = \sum_{x_i, x_j \in C_i} dist\ (x_i, x_j)$$

Usando os resultados encontrados na modelagem matemática do algoritmo, os seguintes *clusters* foram encontrados:

$$C_1 = \{x_1, x_5, x_6\}$$
$$C_2 = \{x_2, x_3, x_4\}$$

O *cluster* C_1 possui os pontos x_1, x_5 e x_6, que formam três pares de objetos:

$$C_1 = \{(x_1, x_5), (x_1, x_6), (x_5, x_6)\}$$

Portanto, a quantidade de pares n_w de pontos do *cluster* C_1 é definida como:

$$n_w = 3$$

As distâncias entre esses pares são:

$$dist\ (x_1, x_5) = 0{,}1414$$
$$dist\ (x_1, x_6) = 0{,}2000$$
$$dist\ (x_5, x_6) = 0{,}3162$$

A soma das distâncias entre os pares de pontos do agrupamento C_1 é realizada:

$$d_w(C_1) = 0{,}1414 + 0{,}2000 + 0{,}3162$$
$$d_w(C_1) = 0{,}6576$$

A base de dados *D* utilizada na modelagem matemática do DBSCAN possui 21 pares de pontos:

$$D = \left\{ \begin{matrix} (x_1, x_2), (x_1, x_3), (x_1, x_4), (x_1, x_5), (x_1, x_6), (x_1, x_7), (x_2, x_3), \\ (x_2, x_4), (x_2, x_5), (x_2, x_6), (x_2, x_7), (x_3, x_4), (x_3, x_5), (x_3, x_6), \\ (x_3, x_7), (x_4, x_5), (x_4, x_6), (x_4, x_7), (x_5, x_6), (x_5, x_7), (x_6, x_7) \end{matrix} \right\}$$

Portanto, o passo seguinte consiste em obter as n_w menores distâncias entre esses pares de objetos da base, sendo que para o *cluster* C_1, $n_w = 3$:

$$dist\ (x_1, x_5) = 0{,}1414$$

$$dist\,(x_3, x_4) = 0{,}1414$$
$$dist\,(x_1, x_6) = 0{,}2020$$

Realizando o somatório das n_w menores distâncias os pares de objetos do conjunto de dados:

$$\min(n_w) = 0{,}1414 + 0{,}1414 + 0{,}2020$$
$$\min(n_w) = 0{,}4848$$

As n_w maiores distâncias entre os pares objetos da base de dados, também são obtidas:

$$dist\,(x_4, x_7) = 3{,}4015$$
$$dist\,(x_3, x_7) = 3{,}3061$$
$$dist\,(x_2, x_7) = 3{,}1000$$

Realizando o somatório das n_w maiores distâncias os pares de pontos do conjunto de dados:

$$\max(n_w) = 3{,}4015 + 3{,}3061 + 3{,}1000$$
$$\max(n_w) = 9{,}8076$$

Resolvendo-se o a equação do *C-Index* para o *cluster* C_1 tem-se:

$$C - Index\,(C_1) = \frac{0{,}6576 - 0{,}4848}{9{,}8076 - 0{,}4848}$$
$$C - Index\,(C_1) = \frac{0{,}1728}{9{,}3228}$$
$$C - Index\,(C_1) = 0{,}018715882505848713283077729009617884066$$

Como pode ser verificado, o valor calculado para o *C-Index* do *cluster* C_1 apresentou um valor baixo, bastante próximo de zero, o que indica que esse grupo possui uma boa coesão interna, contendo pontos bastante similares entre si.

O *cluster* C_2 possui os pontos x_2, x_3 e x_4, que formam três pares de objetos:

$$C_1 = \{(x_2, x_3), (x_2, x_4), (x_3, x_4)\}$$

A quantidade de pares n_w de pontos do *cluster* C_2 é definida:

$$n_w = 3$$

As distâncias os pares de pontos do *cluster* são:

$$dist\,(x_2, x_3) = 0{,}2828$$
$$dist\,(x_2, x_4) = 0{,}3162$$
$$dist\,(x_3, x_4) = 0{,}1414$$

A soma das distâncias entre esses pares é realizada:

$$d_w(C_2) = 0{,}2828 + 0{,}3162 + 0{,}1414$$
$$d_w(C_1) = 0{,}6576$$

A base de dados *D* utilizada na modelagem matemática do DBSCAN possui 21 pares de pontos:

$$D = \left\{ \begin{matrix} (x_1, x_2), (x_1, x_3), (x_1, x_4), (x_1, x_5), (x_1, x_6), (x_1, x_7), (x_2, x_3), \\ (x_2, x_4), (x_2, x_5), (x_2, x_6), (x_2, x_7), (x_3, x_4), (x_3, x_5), (x_3, x_6), \\ (x_3, x_7), (x_4, x_5), (x_4, x_6), (x_4, x_7), (x_5, x_6), (x_5, x_7), (x_6, x_7) \end{matrix} \right\}$$

O passo seguinte consiste em obter as n_w menores distâncias entre esses pares de objetos da base, sendo que para o *cluster* C_2, $n_w = 3$:

$$dist\,(x_1, x_5) = 0{,}1414$$
$$dist\,(x_3, x_4) = 0{,}1414$$
$$dist\,(x_1, x_6) = 0{,}2020$$

Realizando o somatório das n_w menores distâncias os pares de objetos do conjunto de dados:

$$\min(n_w) = 0{,}1414 + 0{,}1414 + 0{,}2020$$
$$\min(n_w) = 0{,}4848$$

As n_w maiores distâncias entre os pares objetos da base de dados são obtidas:

$$dist\,(x_4, x_7) = 3{,}4015$$
$$dist\,(x_3, x_7) = 3{,}3061$$
$$dist\,(x_2, x_7) = 3{,}1000$$

Realizando o somatório das n_w maiores distâncias os pares de pontos do conjunto de dados:

$$\max(n_w) = 3{,}4015 + 3{,}3061 + 3{,}1000$$
$$\max(n_w) = 9{,}8076$$

Resolvendo-se o a equação do *C-Index* para o *cluster* C_2:

$$C - Index(C_2) = \frac{0{,}7404 - 0{,}4848}{9{,}8076 - 0{,}4848}$$
$$C - Index(C_2) = \frac{0{,}2556}{9{,}3228}$$
$$C - Index(C_2) = 0{,}027416655940275453726348307375466598017 8$$

Para o agrupamento C_2 o índice também apresentou um valor baixo, o que indica que esse grupo possui uma boa coesão interna, sendo bastante homogêneo com relação aos seus objetos.

6.2.4 Implementação e Realização de Testes

O algoritmo DBSCAN foi desenvolvido no módulo de clusterização da *Shell Orion Data Mining Engine*, por meio da linguagem de programação Java e do ambiente de programação integrado NetBeans 7.0.1[22].

A *Shell Orion* possibilita que sejam estabelecidas conexões com diversos Sistemas Gerenciadores de Bancos de Dados (SGBD), contanto que esses SGBD disponibilizem um *driver* JDBC para realizar a conectividade. Já se encontram a disposição do usuário os seguintes SGBD: *PostgreSQL*, HSQLDB, *Firebird*, *Oracle Express Edition*, *MySQL* e *SQLAnywhere*.

Na implementação e realização de testes do algoritmo DBSCAN, o SGBD escolhido foi o HSQLDB[23], por não necessitar de instalação prévia, ser gratuito e de simples portabilidade.

[22] Disponível para download gratuitamente em (http://netbeans.org).
[23] O HSQLDB é um SGBD relacional totalmente escrito em Java e está disponível para download gratuitamente em (http://hsqldb.org).

Após a definição do SGBD, testes foram realizados com uma base de dados específica para verificar os resultados obtidos pelo algoritmo desenvolvido. A base de dados escolhida para a realização desses testes foi de indicadores ambientais das bacias hidrográficas da região carbonífera catarinense.

Portanto, com os dados devidamente pré-processados e normalizados, deve-se realizar a inserção dos mesmos no HSQLDB por meio de comandos SQL, e após isso é realizada a conexão da *Shell Orion* com o SGBD no menu *Arquivo*, submenu *Conectar*.

A partir do momento em que é estabelecida uma conexão com um SGBD, torna-se possível acessar o algoritmo DBSCAN por meio do menu *Data Mining* e dos submenus *Clusterização*, *Densidade*, algoritmo DBSCAN (Figura 20).

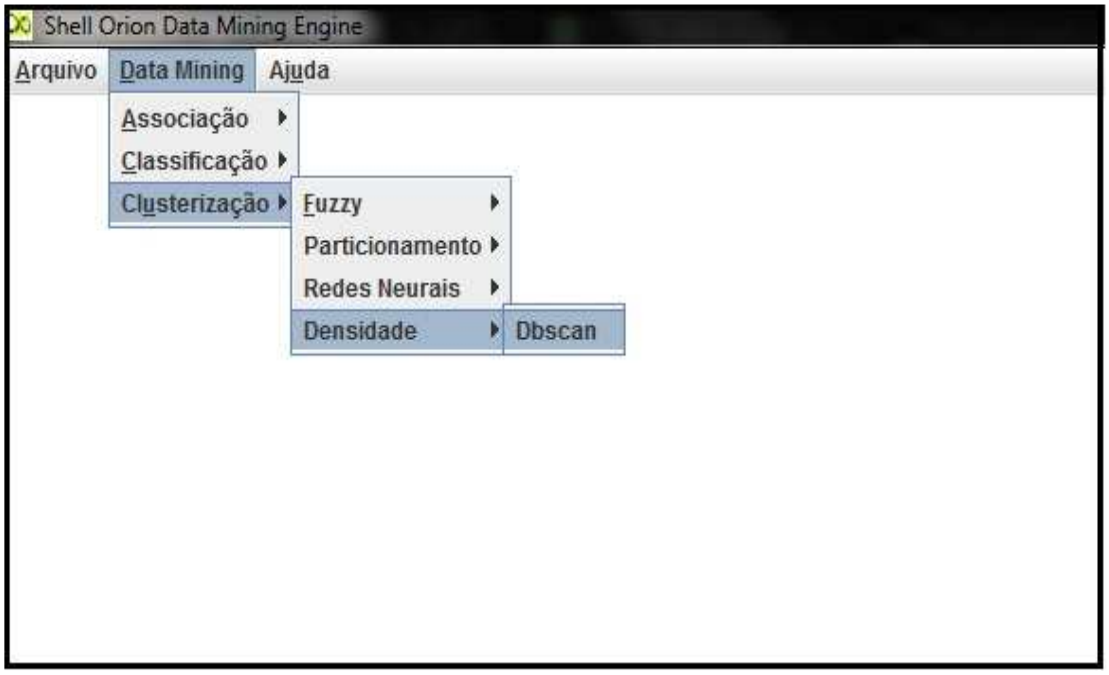

Figura 20. Acesso ao menu do algoritmo DBSCAN

A tarefa de clusterização por meio do algoritmo DBSCAN requer que o usuário informe dois parâmetros de entrada. Esses parâmetros devem ser informados no quadrante superior esquerdo denominado *Parâmetros do Algoritmo* na tela inicial do DBSCAN (Figura 21).

Os parâmetros solicitados ao usuário são os seguintes:

a) **vizinhança ε (*épsilon*):** determina o raio de vizinhança ε para cada ponto da base de dados. Dado o parâmetro ε, o algoritmo DBSCAN verifica a quantidade de pontos contidos no raio ε para cada ponto da base de dados, e se essa quantidade exceder certo número, um *cluster* é formado;

b) **número mínimo de pontos (η):** parâmetro que especifica o número mínimo de pontos que certo objeto da base de dados necessita possuir na ε-vizinhança para ser considerado um ponto central e consequentemente, de acordo com as definições de *cluster* baseado em densidade, formar um *cluster* agregando todos os pontos vizinhos.

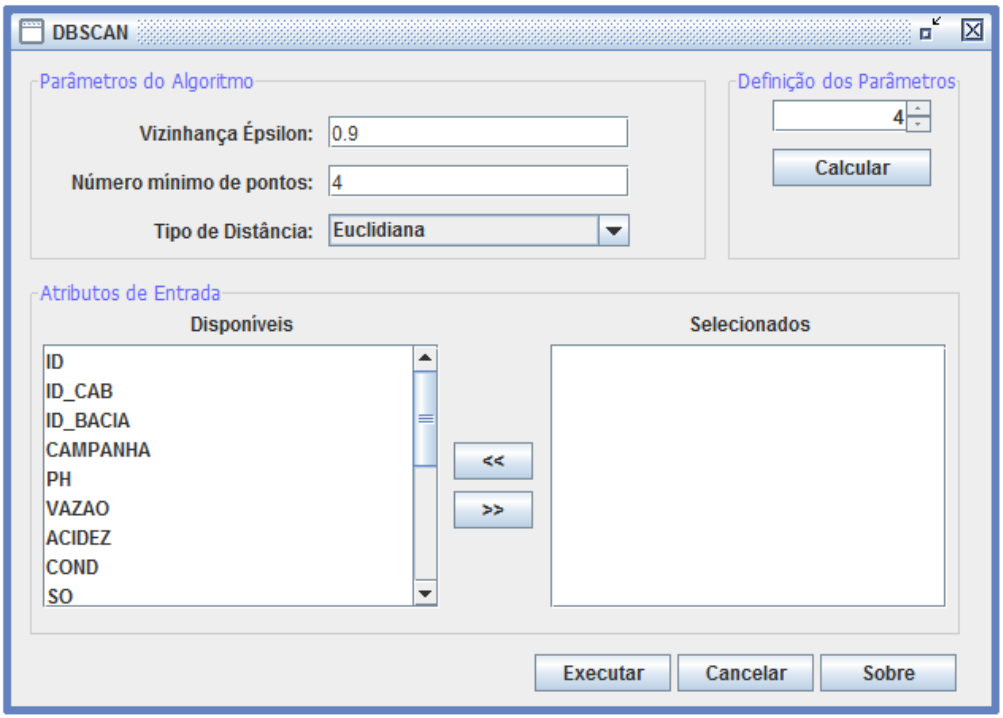

Figura 21. Seleção dos parâmetros de entrada para o algoritmo DBSCAN

Como a definição dos parâmetros de entrada não é uma tarefa tão simples de ser realizada, no algoritmo atribui-se para bases de dados bidimensionais, o valor padrão 4 para o parâmetro η. Valores de η maiores que este, segundo os autores do algoritmo, não geram resultados significantemente diferentes, além de necessitarem de computação extra.

Assim, para o parâmetro η usa-se o valor padrão 4 e para o parâmetro ε defini-se

o valor de acordo o método heurístico denominado função *k-dist* com $k = \eta$, ou seja, a função é executada como *4-dist*.

Na execução da função *k-dist*, deve ser selecionada uma medida de distância na tela principal do algoritmo, bem como os atributos de entrada desejados. Após isso, no quadrante superior da direta, denominado *Definição dos Parâmetros*, na tela principal do algoritmo (Figura 21), deve se informar o valor para *k*, caso se deseje um número diferente do valor padrão 4. Finalmente, solicita-se a *Shell Orion* o cálculo da função *k-dist* (Figura 22).

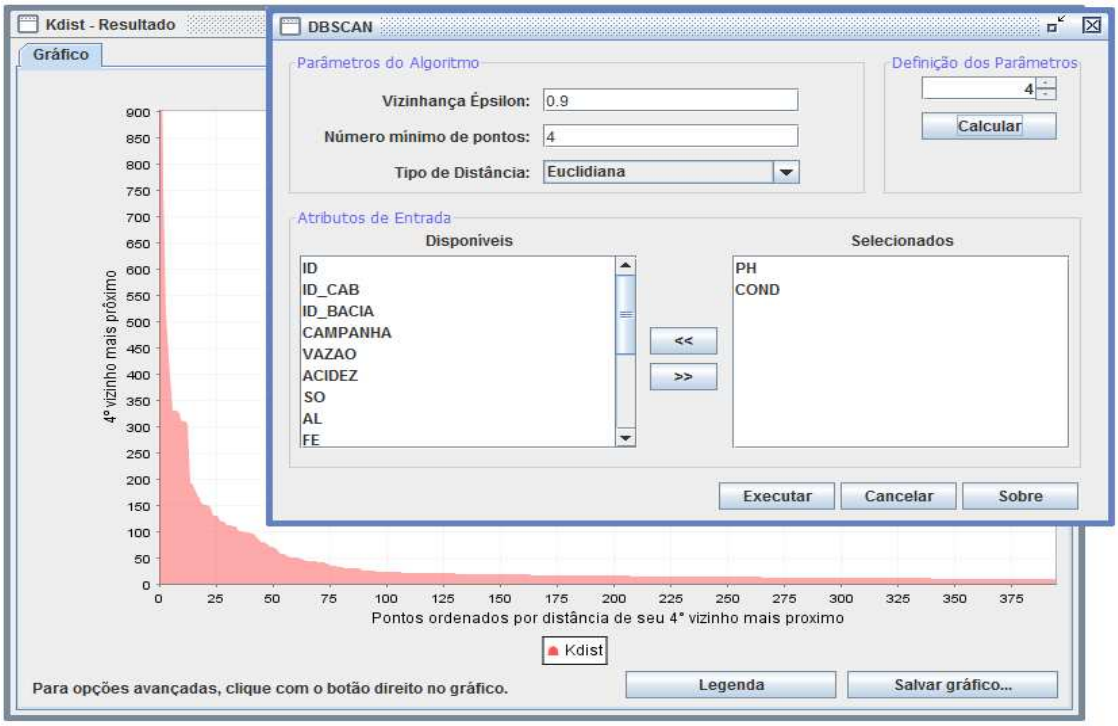

Figura 22. Heurística para auxilio na definição do parâmetro ε

O gráfico da função *k-dist* (Figura 23) traz todos os pontos da base de dados ordenados decrescentemente em função da distância do seu *k*-ésimo vizinho mais próximo. Com esse gráfico é possível obter dicas de como se comporta a distribuição de densidade no conjunto de dados e assim ele pode ser útil para ajudar o usuário na definição do parâmetro ε.

Visualizando esse gráfico, o usuário poderá verificar no eixo *y* (ordenadas) que

os valores apresentados dizem respeito às distâncias ordenadas decrescentemente entre um ponto e seu *k*-ésimo vizinho mais próximo. No eixo *x* (abscissas) estão os pontos da base de dados.

Analisando-se a Figura 23 é possível verificar que no eixo *y* entre os valores 0,04 e 0,06 ocorre repentinamente uma mudança brusca no gráfico. Aproximadamente entre esses valores é possível identificar um “vale” que pode ser definido como o *cluster* menos denso da base de dados. Portanto, todos os valores a direita desse “vale” estão contidos em algum *cluster* e todos os pontos mais a esquerda desse ponto podem ser considerados pontos de borda e *outliers*.

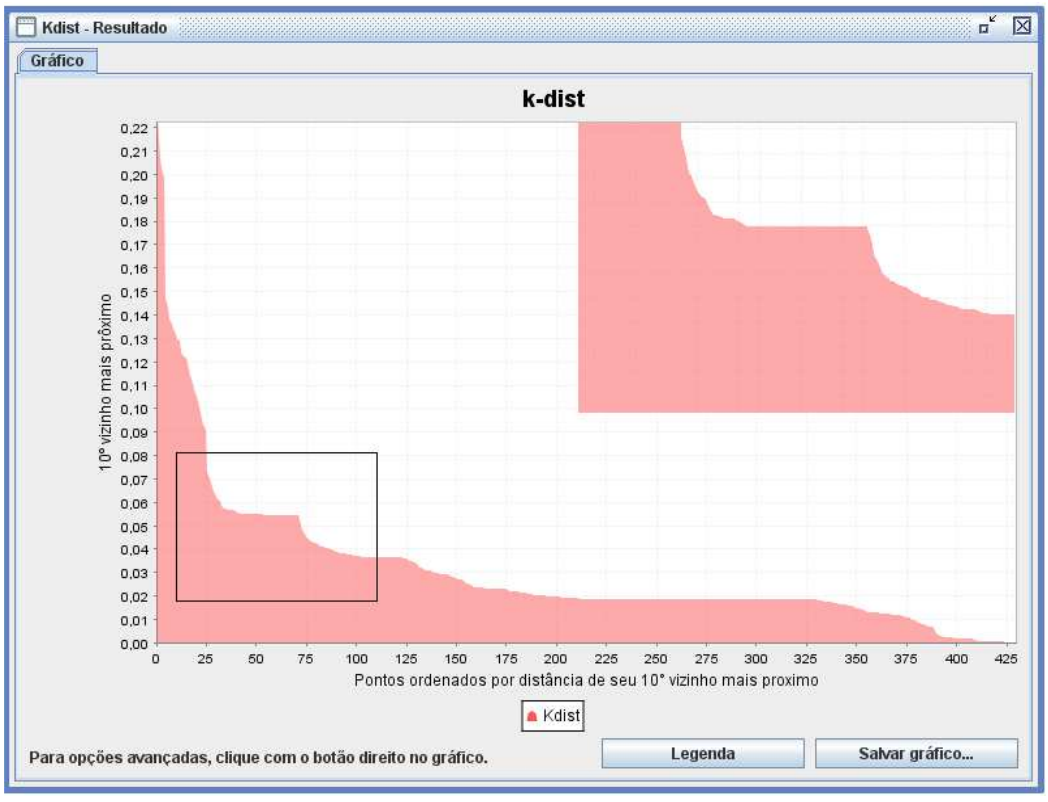

Figura 23. Gráfico da função *k-dist*

De acordo com o gráfico (Figura 23) é possível definir para o parâmetro ε valores entre 0,04 e 0,06, pois se verifica que no eixo *y* (ordenadas) aproximadamente entre esses valores, ocorre o primeiro “vale” e esses valores representam o *cluster* menos denso da base de dados. Como a função *k-dist* foi executada usando $k = 10$, então se define para o

parâmetro η o valor 10 e para o parâmetro ε um valor de aproximadamente 0,05.

É importante salientar que por padrão o algoritmo sempre recomenda o valor 4 para η, porém em alguns casos, principalmente quando o conjunto de dados possui *clusters* com densidades muito distintas, esse pode não ser o valor apropriado, e portanto deve-se testar a execução do algoritmo com outros valores para esse parâmetro.

Os valores selecionados para os parâmetros de entrada interferem diretamente nos resultados gerados pelo algoritmo e devem ser cuidadosamente selecionados para se obter bons resultados e grupos bem definidos.

Valores altos demais para o parâmetro ε tendem a classificar todos os pontos da base de dados somente em um único *cluster*, enquanto valores pequenos demais podem classificar todos os pontos como *outliers* e não encontrar nenhum *cluster*.

A *Shell Orion* possibilita que os resultados obtidos sejam analisados por meio de resumo, árvore e gráfico, além de também permitir que o usuário possa exportar o resultado obtido pela clusterização para um arquivo SQL. No caso do DBSCAN, os dados classificados como *outliers* também são incluídos neste arquivo.

Na Figura 24 os resultados são demonstrados por meio de um resumo textual contendo informações sobre o tempo de execução do algoritmo, atributos de entrada utilizados e a classificação dos pontos da base de dados, ou seja, o *cluster* em que cada ponto está inserido, ou se ele é um *outlier*. Alguns índices de validação que buscam avaliar a qualidade das partições encontradas pelo DBSCAN também podem ser visualizados.

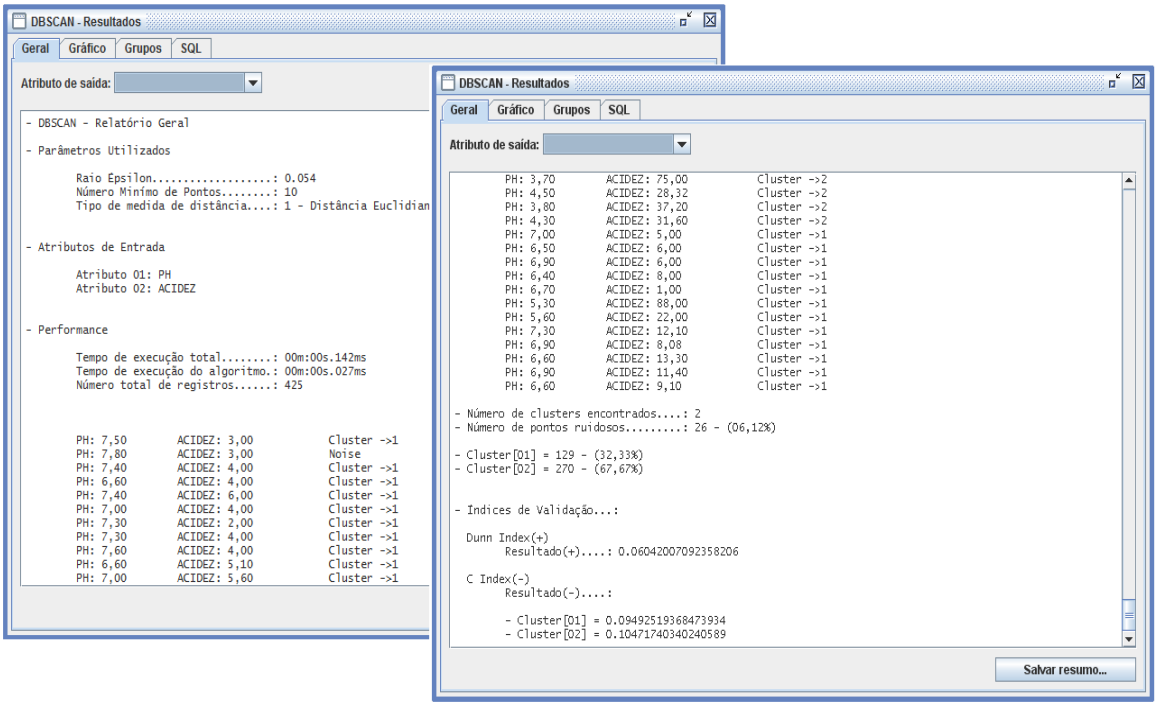

Figura 24. Resumo textual da clusterização por meio do algoritmo DBSCAN

Por padrão a caixa *Atributo de Saída*, é apresentada sem nenhum atributo selecionado, porém, caso o usuário deseje selecionar um atributo de saída específico, o resumo é apresentado de modo detalhado (Figura 25).

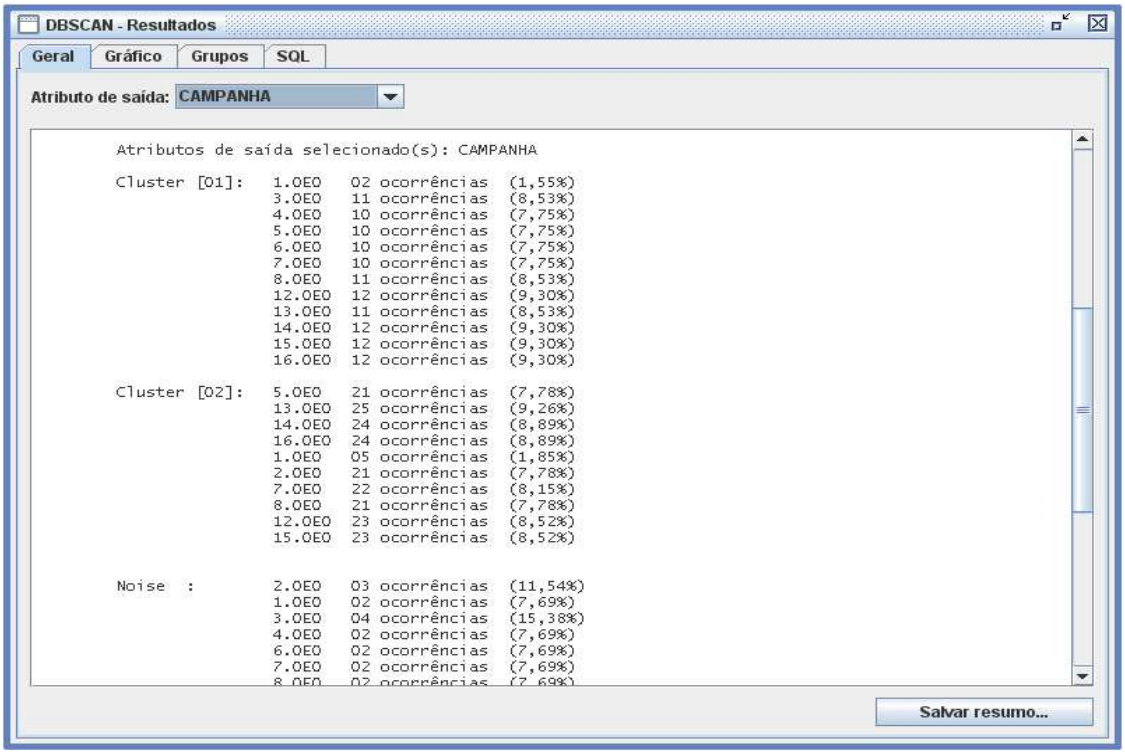

Figura 25. Resumo da clusterização com atributo de saída selecionado

A distribuição dos dados e a identificação dos *clusters* produzidos também podem ser visualizados em forma gráfica (Figura 26). O gráfico é gerado por meio da técnica *Principal Component Analysis* (PCA). Essa técnica utiliza sucessivas decomposições nos dados, com o objetivo de diminuir a dimensionalidade dos dados originais, permitindo assim que esses dados possam ser projetados em um gráfico.

Em uma PCA, a dimensão original dos dados é reduzida para um conjunto de dimensões denominadas Componentes Principais (PC). A partir dos PC são gerados dois novos conjuntos de dados chamados *scores* e *loadings*, que possuem, respectivamente, informações sobre as amostras e as variáveis. Ao se combinar os dados dos *scores* é possível efetuar de maneira mais criteriosa um estudo dos dados originais sem a perda de informações relevantes (MARTENS; NAES, 1993, tradução nossa).

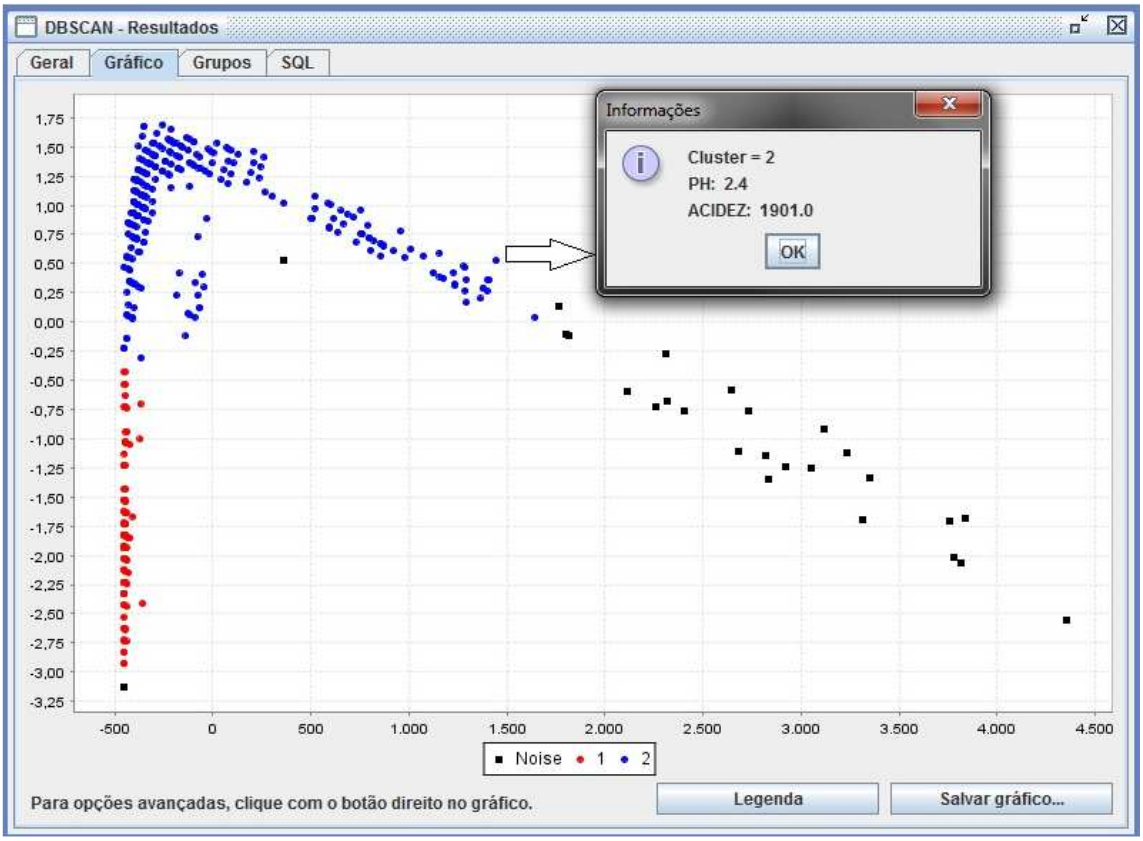

Figura 26. Gráfico construído por meio da PCA para o algoritmo DBSCAN

Também é possível analisar o resultado gerado pelo algoritmo por meio de uma

estrutura de árvore (Figura 27). Essa estrutura permite a visualização detalhada dos resultados gerados pelo algoritmo DBSCAN.

Com o objetivo de otimizar o desempenho e melhorar o tempo de execução total do módulo desenvolvido, a árvore não é gerada automaticamente pela *Shell Orion*. Caso o usuário deseje visualizar essa estrutura, deve marcar o campo *Gerar Árvore* na tela referente à árvore.

Os resultados gerados pelo algoritmo DBSCAN também podem ser exportados para o formato SQL (Figura 28). Essa funcionalidade permite que o resultado da clusterização seja usado posteriormente em outras tarefas, como por exemplo, a classificação e a análise de *outliers*, uma vez que são exportados para uma tabela específica os pontos classificados como *outliers* pelo DBSCAN.

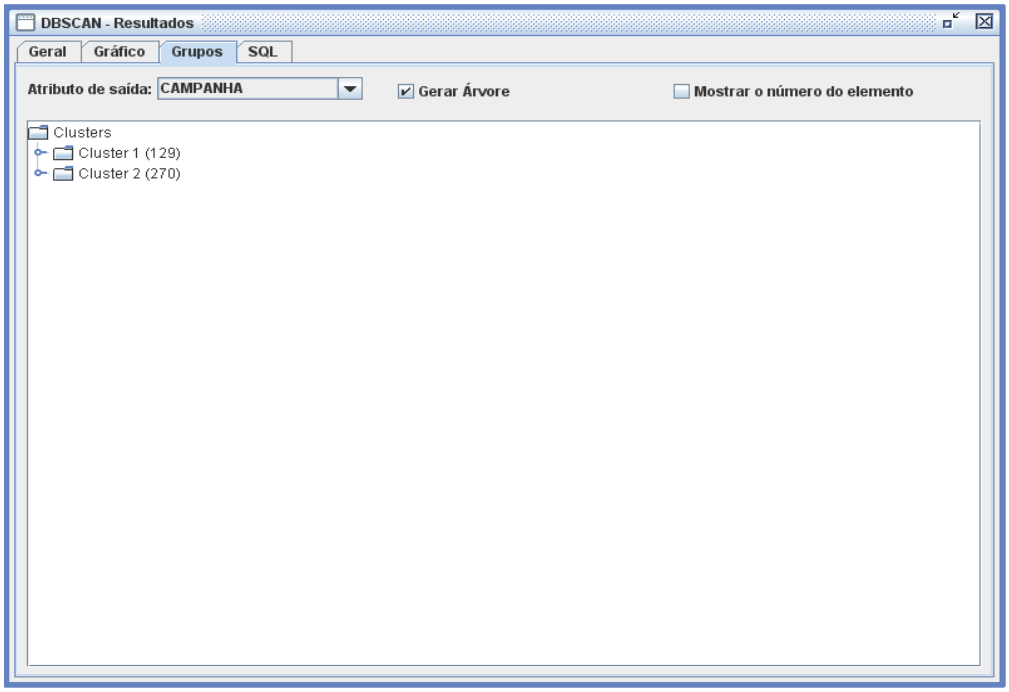

Figura 27. Análise dos resultados por meio da estrutura de árvore

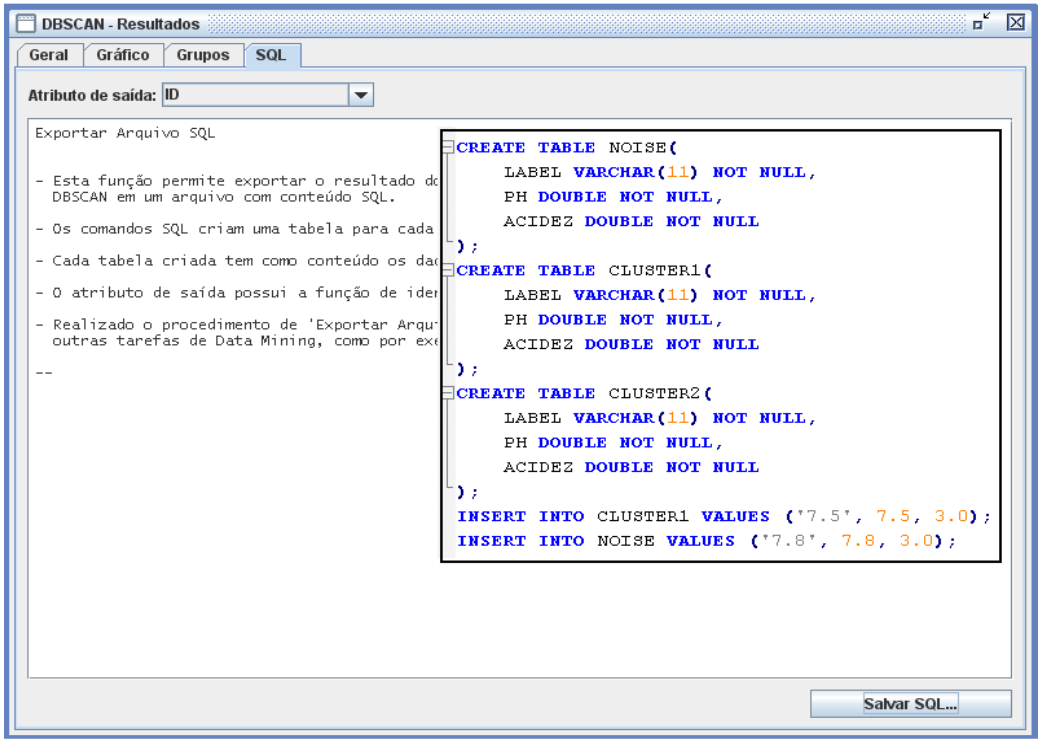

Figura 28. Exportação dos resultados para o formato SQL

Após a finalização da implementação, foram efetuados testes com o objetivo de comprovar os resultados corretos e verificar os tempos de execução do algoritmo, sendo que resultados obtidos e a discussão desses resultados são apresentados a seguir.

6.2.5 Análise dos Dados

O propósito fundamental de uma análise é organizar os dados de modo que permitam responder ao problema colocado. Costumeiramente, inicia-se qualquer análise de dados por uma descrição das variáveis observadas, incluindo-se medidas de tendência central e variabilidade (BSQUERRA; SARRIERA; MARTINEZ, 2004).

Na análise estatística da comparação dos tempos de processamento obtidos pelo algoritmo DBSCAN implementado na *Shell Orion* em comparação com a ferramenta *Weka* as seguintes etapas foram realizadas:

a) **medidas de tendência central e variabilidade:** esse tipo de medida permite

um conhecimento inicial dos dados em análise. Foram utilizadas medidas de tendência central, tais como a média e a mediana, pois permitem estimar o valor real do que se está analisando. Também foram empregadas algumas medidas de dispersão das amostras tais como: amplitude, variância e desvio padrão, pois essas medidas avaliam a dispersão do conjunto de valores que estão sendo estudados (BARBETTA; REIS; BORNIA, 2010);

b) **verificação e remoção de *outlliers*:** valores atípicos podem distorcer uma análise de dados, tornando o estudo tendencioso. Portanto, com o objetivo de identificar os *outliers* nas amostras de tempo obtidas com a execução do algoritmo DBSCAN, foi utilizado o gráfico *boxplot*, por permitir avaliar de maneira gráfica e intuitiva, a simetria dos dados, sua dispersão e a existência de *outliers* nos mesmos (BARBETTA; REIS; BORNIA, 2010; FIELD, 2009);

c) **teste de normalidade da distribuição das amostras:** após a identificação dos *outliers* por meio do gráfico *boxplot*, o teste de *Kolmogorov-Smirnov* foi utilizado para testar a hipótese de normalidade dos dados. Esse teste foi utilizado, pois verifica se os dados presentes em uma amostra comportam-se de acordo com uma distribuição teórica, a normal no caso. A verificação da normalidade da distribuição é importante, pois a partir dessa averiguação é decidido qual o teste estatístico que será utilizado para a comparação dos resultados, visto que diversos testes estatísticos de comparação de médias, como o teste *T* de *Student*, exigem que os dados sejam provenientes de uma distribuição aproximadamente normal (BARBETTA; REIS; BORNIA, 2010; BISQUERRA; SARRIERA; MARTINEZ, 2004);

d) **teste não-paramétrico *U* de *Mann-Whitney*:** para a realização desse teste

> não-paramétrico de comparação de médias é desnecessária a especificação da distribuição da população de onde provém a amostra, ou seja, ele é aplicável independentemente da distribuição da população. Como os tempos de processamento coletados não seguiram uma distribuição normal, esse teste foi utilizado. O teste *U* de *Mann-Whitney* é utilizado para a verificação da hipótese de diferença estatística entre as médias de tempo de processamento, pois se trata de um dos principais testes não-paramétricos para comparação de médias, requerendo poucos pressupostos acerca dos dados, sendo o teste não-paramétrico equivalente ao teste *T* de *Student* para amostras independentes (BARBETTA; REIS; BORNIA, 2010; BISQUERRA; SARRIERA; MARTINEZ, 2004).

A aplicação de testes estatísticos permitiu a melhor compreensão dos resultados obtidos nos testes a que foram submetidos o algoritmo DBSCAN, evidenciando as supostas diferenças existentes entre os tempos de processamento nas ferramentas comparadas. No Apêndice D estão descritos de maneira mais detalhada alguns dos conceitos estatísticos estudados no trabalho.

6.3 RESULTADOS OBTIDOS

Os testes realizados compreenderam a análise dos *clusters* gerados pelo algoritmo por meio de índices de validação, a análise de desempenho do algoritmo com relação ao tempo de execução e a comparação do algoritmo implementado na *Shell Orion* com o desenvolvido na ferramenta *Weka*, sendo que para todos os testes a base de dados de monitoramento de indicadores ambientais foi utilizada.

Na realização dos testes com o algoritmo DBSCAN foi utilizado um

microcomputador com sistema operacional Windows7, processador Intel Core i5 2.53 GHz e 4GB de memória RAM.

6.3.1 *Clusters* Encontrados pelo Algoritmo DBSCAN

Na avaliação do algoritmo DBSCAN implementado na *Shell Orion*, utilizou-se uma base de dados da área de engenharia ambiental referente ao monitoramento de bacias hidrográficas da região carbonífera.

Nesta pesquisa analisou-se os dados de monitoramento da bacia do rio Araranguá, compostos por 679 registros coletados ao longo de 69 pontos de monitoramento espalhados por essa bacia.

Na seleção dos parâmetros de entrada do algoritmo adotou-se o seguinte critério:

a) os atributos de entrada desejados foram selecionados;
b) para o parâmetro η foram selecionados valores entre 4 e 10;
c) a função *k-dist* foi calculada iterativamente com *k* valendo η em cada iteração, ou seja η*-dist*;
d) o parâmetro ε foi definido iterativamente analisando-se o gráfico *k-dist*;
e) os mesmos passos foram repetidos usando as três medidas de distância implementadas no trabalho: euclidiana, euclidiana normalizada e *Manhattan*.

Os atributos de entrada selecionados para o primeiro teste foram os índices de pH e a concentração de ferro presentes nas amostras obtidas ao longo da bacia do rio Araranguá. O objetivo desse teste foi separar os registros coletados em dois grupos, ou seja, pontos de coleta em que a água está contaminada e pontos de coleta em que a água não está contaminada.

Na Tabela 8 verifica-se os parâmetros de entrada selecionados e que foram

usados pelo algoritmo DBSCAN para cada medida de distância.

Tabela 8. Parâmetros de entrada para os atributos pH e concentração de ferro

Tipo de distância	Número mínimo de pontos (η)	ε-vizinhança
Euclidiana	4	170
Euclidiana Normalizada	8	0, 057
Manhattan	6	150

Na Tabela 9 é possível verificar os resultados gerados pelo algoritmo DBSCAN usando os atributos de entrada pH e concentração de ferro com a medida de distância euclidiana normalizada.

Tabela 9. *Clusters* encontrados usando a distância euclidiana normalizada

Cluster	Quantidade de elementos	Porcentagem	Situação do ponto de coleta
1	132	19,44%	não-contaminado
2	538	79,23%	contaminado
Outliers	9	01,33%	-

Empregando-se a distância euclidiana normalizada o DBSCAN identificou que em 79,23% das amostras coletadas na bacia do rio Araranguá os índices de pH estão geralmente abaixo de 5.0 e a concentração de ferro presente nas águas se encontra em índices geralmente maiores que 20 mg/L. De acordo com Alexandre e Krebs (1996) baixos valores de pH e altas concentrações de ferro e outros sulfatos demonstram a degradação dos rios da bacia por atividades ligadas a extração de carvão e fazem com que os seus recursos hídricos se apresentem de maneira imprópria para o consumo humano e uso em geral.

Na Figura 29 é possível verificar os resultados obtidos pelo algoritmo, onde se percebem claramente os dois *clusters* encontrados pelo algoritmo. Alguns *outliers* foram identificados, mostrando que o DBSCAN é robusto na presença desse tipo de dado. Também pode ser visto que foram encontrados dois agrupamentos com formas diferentes, demonstrando a capacidade que o algoritmo possui de identificar grupos com formas arbitrárias.

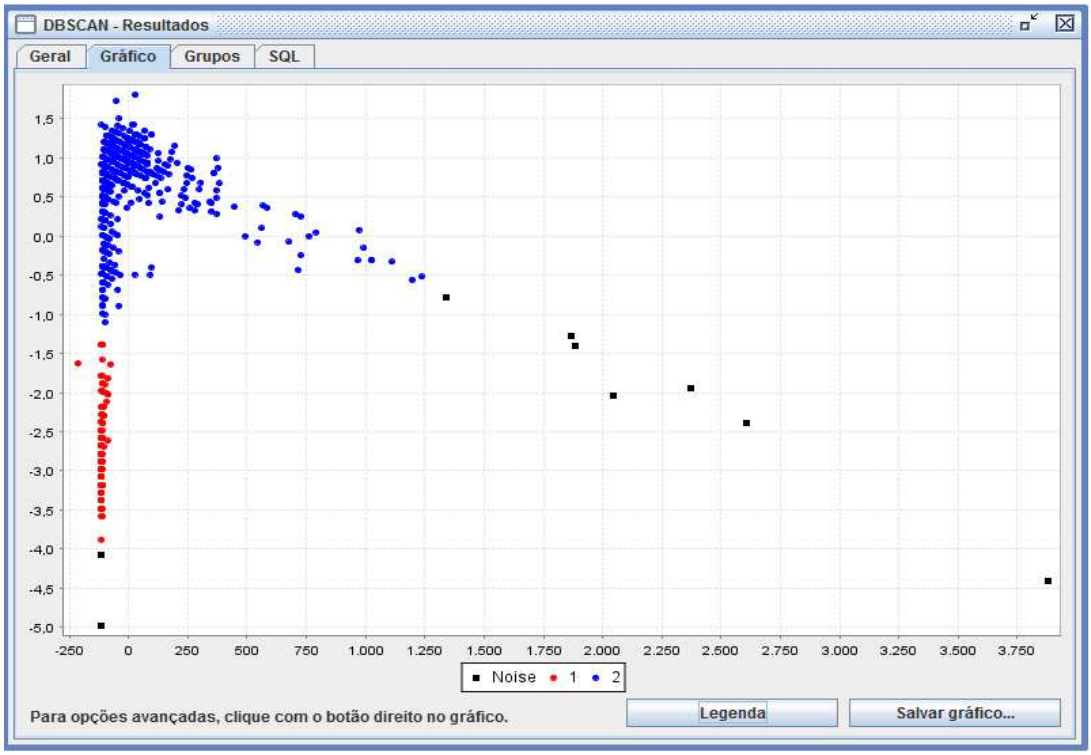

Figura 29. Resultados obtidos pelo DBSCAN

Utilizando as medidas de distância euclidiana (Tabela 10) e *Manhattan* (Tabela 11), verificou-se que os resultados encontrados não foram semelhantes aos da distância euclidiana normalizada, sendo que com essas duas medidas o DBSCAN não foi capaz de separar a base de dados de modo satisfatório, gerando um único *cluster* contendo 97,80% dos pontos para as duas medidas. Nesse *cluster* estão misturados pontos de coleta não-contaminados e contaminados. O percentual de *outliers* encontrados foi de 00,88% para a distância euclidiana e 01,03% para a distância *Manhattan*.

Tabela 10. *Clusters* encontrados usando distância euclidiana

Cluster	**Quantidade de elementos**	**Porcentagem**
1	664	97,80%
2	9	01,32%
Outliers	6	00,88%

Tabela 11. *Clusters* encontrados usando distância *Manhattan*

Cluster	**Quantidade de elementos**	**Porcentagem**
1	664	97,80%
2	8	01,17%
Outliers	7	01.03%

O índice de *Dunn* apontou um valor baixo para a clusterização usando distância euclidiana normalizada, porém isso se deve a dispersão interna de um dos *clusters* encontrados e, esse resultado pode ser melhorado se o parâmetro η for sendo gradativamente aumentado, crescendo assim a densidade intra-*cluster* e tornando os agrupamentos mais compactos e densos. Isso demonstra que os índices de validação são subjetivos e também devem ser avaliados com cuidado. Já o *C-Index* apresentou valores baixos para todos os grupos, indicando uma boa qualidade destes agrupamentos (Tabela 12).

Tabela 12. Índices de validação para os atributos pH e concentração de ferro

Tipo de distância	**Índice de *Dunn***	***C-Index***
Euclidiana	0,1717344017841391	*Cluster* 1 = 0,007206555330476989 *Cluste*r 2 = 0,03874511749076192
Euclidiana Normalizada	0,08672836862839785	*Cluster* 1 = 0,10694296125378311 *Cluste*r 2 = 0,05135642415906838
Manhattan	0,1894109527513337	*Cluster* 1 = 0,007236692853423446 *Cluste*r 2 = 0,030106101360639688

Em uma nova avaliação do algoritmo, foram selecionados como atributos de entrada o pH e a condutividade com o objetivo de separar os registros da base de dados em contaminados e não contaminados.

Novamente os melhores resultados foram obtidos pela distância euclidiana normalizada, sendo que as outras duas funções de distância implementadas não conseguiram definir de maneira clara os *clusters*, classificando em um único *cluster* pontos que poderiam estar em grupos separados.

Utilizando a distância euclidiana normalizada foram encontrados três *clusters* (Figura 30). Nota-se que o DBSCAN encontrou um *cluster* com 10 pontos em que o pH se encontra muito baixo (<2,9) e a condutividade elevada (> 5000 µS/cm), indicando que os pontos de monitoramento onde foram coletados os registros se encontram bastante degradados pela poluição advinda das atividades relacionadas a extração de carvão mineral.

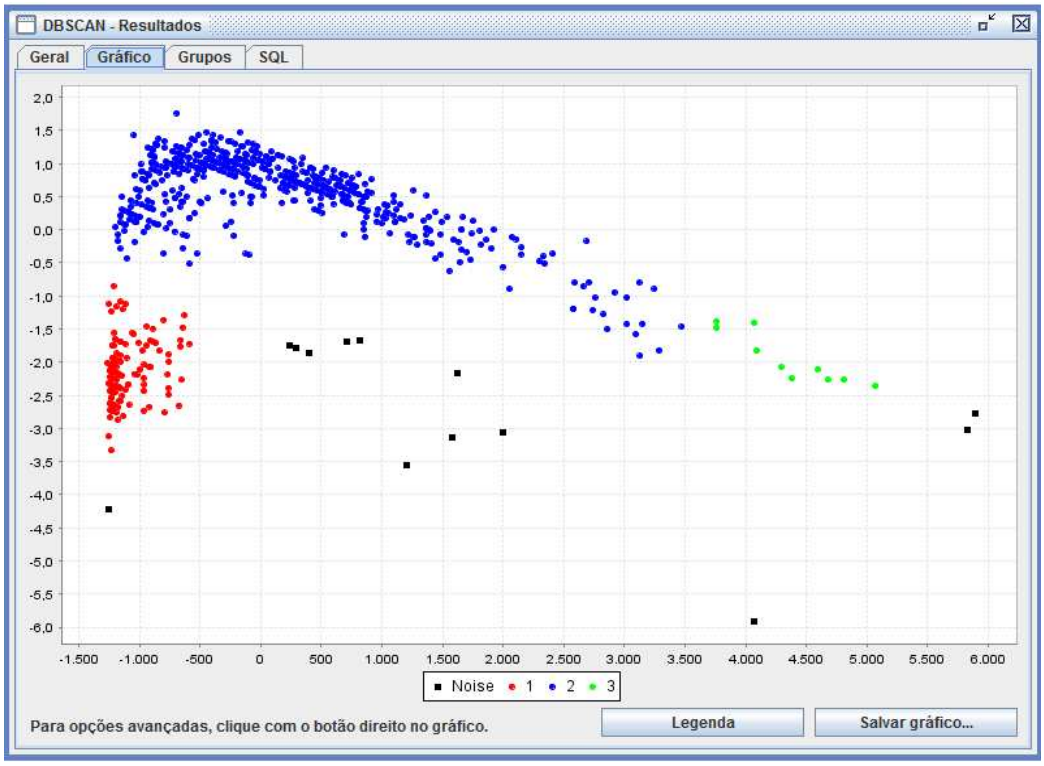

Figura 30. Resultados obtidos usando a distância euclidiana normalizada

Os índices de validação para a clusterização usando os atributos pH e condutividade são apresentados na Tabela 13, onde e possível verificar os valores apresentados.

Tabela 13. Índices de Validação para os atributos pH e condutividade

Tipo de distância	**Índice de *Dunn***	***C-Index***
Euclidiana	0,12589814485377437	*Cluster* 1 = 0,06818941245148834 *Cluster* 2 = 0,03658336497508524
Euclidiana Normalizada	0,0674904037106549	*Cluster* 1 = 0,0839135391967433 *Cluster* 2 = 0,09315682289359811 *Cluster* 3 = 0,06683636708186347
Manhattan	0,046387634312174915	*Cluster* 1 = 0,06874870908065252 *Cluster* 2 = 0,038889511989773286

O índice de *Dunn*, mesmo apresentando um baixo valor para a distância euclidiana normalizada, devido ao fato de que um dos *clusters* encontrados possuir grande dispersão interna e ser pouco compacto indicou uma boa clusterização. Esse índice pode ser melhorado, quando aumentamos a compactação dos *clusters*, por meio de valores maiores para o parâmetro η do algoritmo. Outro fator que torna o valor para esse índice mais baixo

quando é utilizada a distância euclidiana normalizada é que quando essa medida é utilizada, os valores dos atributos são normalizados para o intervalo [0,1] assim resultando em um valor mais baixo para as distâncias entre os pontos da base de dados. O *C-Index* apresentou valores baixos para todos os *clusters* gerados, indicando que o DBSCAN gerou partições com boa coesão interna entre os pontos.

Considerando os diferentes resultados gerados pelo algoritmo DBSCAN, com os mesmos atributos de entrada, porém usando medidas de distância distintas, conclui-se que a escolha dessa medida influencia diretamente no resultado final da clusterização, e portanto deve ser definida de acordo com o resultado esperado e com o tipo de dado presente na base. De modo geral, pode-se dizer que a capacidade do DBSCAN em encontrar agrupamentos, será tão boa quanto à capacidade da medida de distância utilizada pelo algoritmo na análise da base de dados em específico.

6.3.2 Parâmetros de Entrada do Algoritmo DBSCAN

Os parâmetros de entrada do algoritmo também devem ser definidos com precisão, pois valores ligeiramente diferentes para esses parâmetros tendem a gerar resultados finais distintos. Nessa análise empregou-se a distância euclidiana normalizada, e os atributos de entrada pH e condutividade. No parâmetro η utilizou-se sempre o mesmo valor, pois o objetivo foi verificar a influência do parâmetro ε nos resultados gerados pelo algoritmo. A Tabela 14 ilustra como pequenas variações no parâmetro ε produzem resultados totalmente distintos.

Tabela 14. O parâmetro de entrada ε no resultado gerado pelo algoritmo DBSCAN

Número mínimo de pontos (*η*)	***ε*-vizinhança**	**Quantidade de *clusters***	***Quantidade* de *outliers***
4	0,01	43	286
4	0,02	7	80
4	0,03	5	41
4	0,04	3	27
4	0,05	4	16
4	0,06	2	14
4	0,07	2	13
4	0,08	2	9
4	0,09	2	8
4	0,10	2	8
4	0,11	2	7
4	0,12	2	5
4	0,13	1	4
4	0,14	1	3
4	0,15	1	1

Constatou-se que para essa base de dados, conforme o raio da ε-vizinhança de cada ponto aumenta, o algoritmo tende a encontrar mais pontos conectados por densidade e classificá-los em um único *cluster*, pois conforme o valor de ε vai sendo aumentado, os grupos tendem a ficar menos densos e mais dispersos (ESTER et al, 1996, tradução nossa).

Também foi avaliado o impacto do parâmetro η no resultado gerado pelo DBSCAN (Tabela 15). A distância usada foi a euclidiana normalizada e os atributos de entrada foram pH e condutividade. O parâmetro ε foi definido com o valor 0,073.

Tabela 15. O parâmetro de entrada η no resultado gerado pelo algoritmo DBSCAN

Número mínimo de pontos (*η*)	***ε*-vizinhança**	**Quantidade de *clusters***	***Quantidade* de *outliers***
4	0,073	3	9
5	0,073	2	13
6	0,073	2	13
7	0,073	3	13
8	0,073	2	23
9	0,073	2	23
10	0,073	2	23
11	0,073	2	23
12	0,073	2	23
13	0,073	2	23
14	0,073	2	24
15	0,073	2	27
16	0,073	2	27
17	0,073	2	39
18	0,073	2	39

O parâmetro de entrada η possui menos influência no resultado final do DBSCAN do que o parâmetro ε, e conforme o número mínimo de pontos η aumenta, os *clusters* tendem a ficar mais compactos e densos, ao mesmo tempo em que a quantidade de *outliers* aumenta consideravelmente.

A definição dos atributos de entrada também influencia diretamente no resultado da clusterização e, portanto devem ser definidos da melhor maneira possível, de modo que o resultado final seja uma clusterização objetiva e que possa agregar conhecimento acerca do conjunto de dados.

Verificado o correto funcionamento do algoritmo DBSCAN, realizaram-se testes com diversos parâmetros diferentes a fim de analisar o tempo de processamento do algoritmo desenvolvido.

6.3.3 Tempos de Processamento do Algoritmo DBSCAN

Nos testes de desempenho foram analisados diversos parâmetros tais como: quantidade de atributos de entrada, medida de distância utilizada e quantidade de registros. Para obtenção de conjuntos com diversos tamanhos a base de dados original foi sendo gradativamente replicada. Nessa replicação, adotou-se um padrão de crescimento geométrico[24] do qual resultaram as diferentes cargas a serem testadas nos ensaios. Esse padrão foi definido utilizando a seguinte equação:

$$C_K = N * 2^{k-1}$$

Onde:

a) C_k: representa o tamanho (em unidades de registro) da carga a ser testada;

[24] Uma progressão geométrica (seqüência geométrica) é uma seqüência de termos onde existe um termo inicial *a*, e cada termo subseqüente é obtido pelo produto do anterior por um valor constante *r* chamada de razão (GERSTING, 1995).

b) *N*: número de registros da base de dados original;

c) *k*: o número de iterações.

O modelo de crescimento geométrico foi adotado para a definição das cargas a serem utilizadas, pois apresenta como uma de suas características a possibilidade de se obter a visão do comportamento de diferentes tamanhos de cargas (centenas a milhares de registros) em um pequeno intervalo de iterações.

Como o subconjunto referente aos pontos de monitoramento da bacia do rio Araranguá possuía originalmente 679 registros, o tamanho das cargas de dados a serem testadas foram definidas conforme a Tabela 16.

Tabela 16. Definição dos tamanhos das cargas de dados

Iteração	Quantidade de registros	Quantidade de atributos
1	679	2
2	1358	2
3	2716	2
4	5432	2
5	10864	2
6	21728	2
7	43456	2
8	86912	2

Como foi considerado somente a quantidade de registros, os parâmetros η e ε foram fixados respectivamente em 4 e 10. É importante salientar que nesses testes não foram considerados os resultados obtidos pela clusterização, considerando-se somente os tempos de processamento.

Na Tabela 17 são verificados os tempos de processamento obtidos pelo algoritmo DBSCAN com relação ao tamanho da base de dados. É possível notar que os tempos obtidos utilizando a distância euclidiana normalizada foram superiores aos alcançados utilizando a distância euclidiana, pelo motivo da normalização das distâncias entre os pontos da base de dados.

Como a distância *Manhattan* não necessita calcular a raiz quadrada das

distâncias entre os atributos, o algoritmo DBSCAN necessitou de menos tempo de processamento, apresentando melhores resultados para essa medida de distância.

Tabela 17. Efeitos da quantidade de registros no processamento do algoritmo DBSCAN

Quantidade de registros	Distância Euclidiana	Distância Euclidiana Normalizada	Distância *Manhattan*
679	00m: 00s: 047ms	00m: 00s: 106ms	00m: 00s: 030ms
1358	00m: 00s: 187ms	00m: 00s: 335ms	00m: 00s: 080ms
2716	00m: 00s: 686ms	00m: 01s: 188ms	00m: 00s: 280ms
5432	00m: 02s: 605ms	00m: 04s: 754ms	00m: 01s: 010ms
10864	00m: 10s: 437ms	00m: 18s: 925ms	00m: 03s: 870ms
21728	00m: 41s: 309ms	01m: 16s: 311ms	00m: 16s: 160ms
43456	02m: 58s: 043ms	05m: 43s: 463ms	01m: 19s: 709ms
86912	13m: 01s: 347ms	21m: 37s: 672ms	05m: 41s: 684ms

A dimensionalidade se refere à quantidade de atributos presentes no conjunto de dados. Quando o número de atributos presentes no conjunto de dados é grande (alta dimensionalidade), ocorre à degradação das técnicas de análise de dados e do processo de descoberta de conhecimento, assim como aumenta o custo computacional (BOTELHO, 2011).

Na Tabela 18 estão descritos os tempos de processamento necessários ao algoritmo DBSCAN com relação à quantidade de atributos selecionados, ou seja, o efeito da dimensionalidade. Os parâmetros ε e η foram respectivamente fixados em 10 e 4, a carga de dados foi obtida por meio da replicação da base de dados de monitoramento de indicadores ambientais, utilizando o modelo de crescimento geométrico com 4 iterações, totalizando 5432 registros.

Tabela 18. Efeitos da dimensionalidade no processamento do algoritmo DBSCAN

Quantidade de atributos	Distância Euclidiana	Distância Euclidiana Normalizada	Distância *Manhattan*
2	00m: 02s: 605ms	00m: 04s: 754ms	00m: 01s: 010ms
3	00m: 03s: 432ms	00m: 08s: 097ms	00m: 01s: 092ms
4	00m: 04s: 571ms	00m: 09s: 391ms	00m: 01s: 232ms
8	00m: 08s: 643ms	00m: 14s: 242ms	00m: 01s: 794ms
16	00m: 16s: 239ms	00m: 23s: 415ms	00m: 02s: 793ms

Observou-se que o fator dimensionalidade influencia no tempo de processamento do algoritmo de maneira considerável. Novamente a medida de distância que

dispendeu menos tempo foi a *Manhattan*. A distância euclidiana normalizada, que apresentou os melhores resultados em relação à qualidade da clusterização, obteve os piores desempenhos em relação ao tempo, devido ao fato de que todos os atributos são normalizados antes de se calcular a distância propriamente dita, além de ter a necessidade de obter a raiz quadrada das diferenças entre dois atributos da base de dados.

Algumas considerações podem ser feitas em relação aos resultados obtidos no que se refere ao tempo de processamento do algoritmo DBSCAN implementado:

a) **medida de distância**: a função de distância deve ser escolhida de maneira cuidadosa, pois dependendo do tipo de dado existente no conjunto e da quantidade de objetos;

b) **quantidade de atributos**: devem ser escolhidos somente os atributos mais relevantes para a definição dos *clusters*, pois quanto maior a dimensionalidade, mais tempo de processamento deve ser exigido e a qualidade da clusterização pode ser prejudicada de modo considerável;

c) **quantidade de registros**: fator que influencia diretamente o tempo de processamento do algoritmo de clusterização DBSCAN. Bases de dados do mundo real podem possuir milhões de registros, portanto, antes da clusterização, devem passar por um pré-processamento rigoroso, visto que podem existir muitos *outliers* e dados irrelevantes para a tarefa de clusterização.

Após a verificação de desempenho do algoritmo DBSCAN, foi feita uma comparação da *Shell Orion* com a ferramenta *Weka* 3.6 que também possui o algoritmo DBSCAN disponível.

6.3.4 Comparação coma Ferramenta *Weka* 3.6

Com a finalidade de comparar os resultados obtidos pelo DBSCAN na *Shell Orion Data Mining Engine*, foi utilizada a ferramenta *Weka*[25] em sua versão 3.6. A *Waikato Environment for Knowledge Analysis* (Weka) é uma ferramenta distribuída sobre os termos da GNU[26], desenvolvida na Universidade de Waikato, na Nova Zelândia, amplamente utilizada em pesquisas na área de *data mining* e implementa diversos algoritmos de *Data Mining* em Java, o que padroniza as interfaces, facilita a inclusão de novas funcionalidades na ferramenta e permite a sua execução em diversas plataformas (WITTEN; FRANK; HALL, 2011, tradução nossa).

Os seguintes parâmetros de entrada foram utilizados:

a) **ε-vizinhança**: 0,057;

b) **número mínimo de pontos(η)**: 8;

c) **medida de distância**: euclidiana normalizada. A *Weka* possui somente a distância euclidiana normalizada disponível para clusterização por meio do algoritmo DBSCAN;

d) **atributos de entrada**: pH e vazão da água(l/s).

Os resultados (Tabela 19) demonstram que as duas ferramentas obtiveram resultados idênticos.

Tabela 19. Testes de comparação entre o DBSCAN na Shell Orion e na Weka 3.6

Ferramenta	*Cluster* 1	*Cluster* 2	Quantidade de *outliers*
Shell Orion	128 (18,85%)	531 (78,20%)	20 (2,95%)
Weka 3.6	128 (18,85%)	531 (78,20%)	20 (2,95%)

Na Figura 31 é possível verificar os resultados obtidos pelo algoritmo DBSCAN na *Shell Orion Data Mining Engine*.

[25] Disponibilizada gratuitamente em (http://www.cs.waikato.ac.nz/~ml/weka/)
[26] *General Public License*. Informações em (http://www.gnu.org)

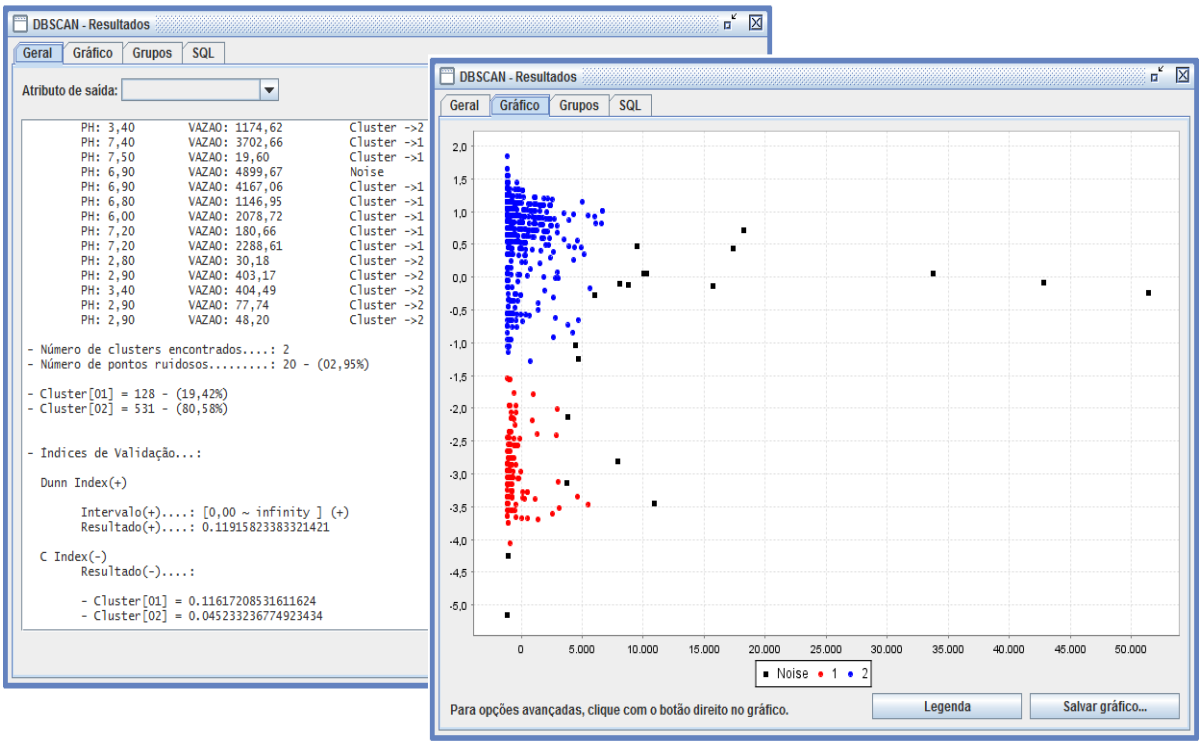

Figura 31. Resultados obtidos pelo DBSCAN na *Shell Orion*

Na Figura 32 tem-se os resultados obtidos pelo algoritmo DBSCAN na ferramenta *Weka*.

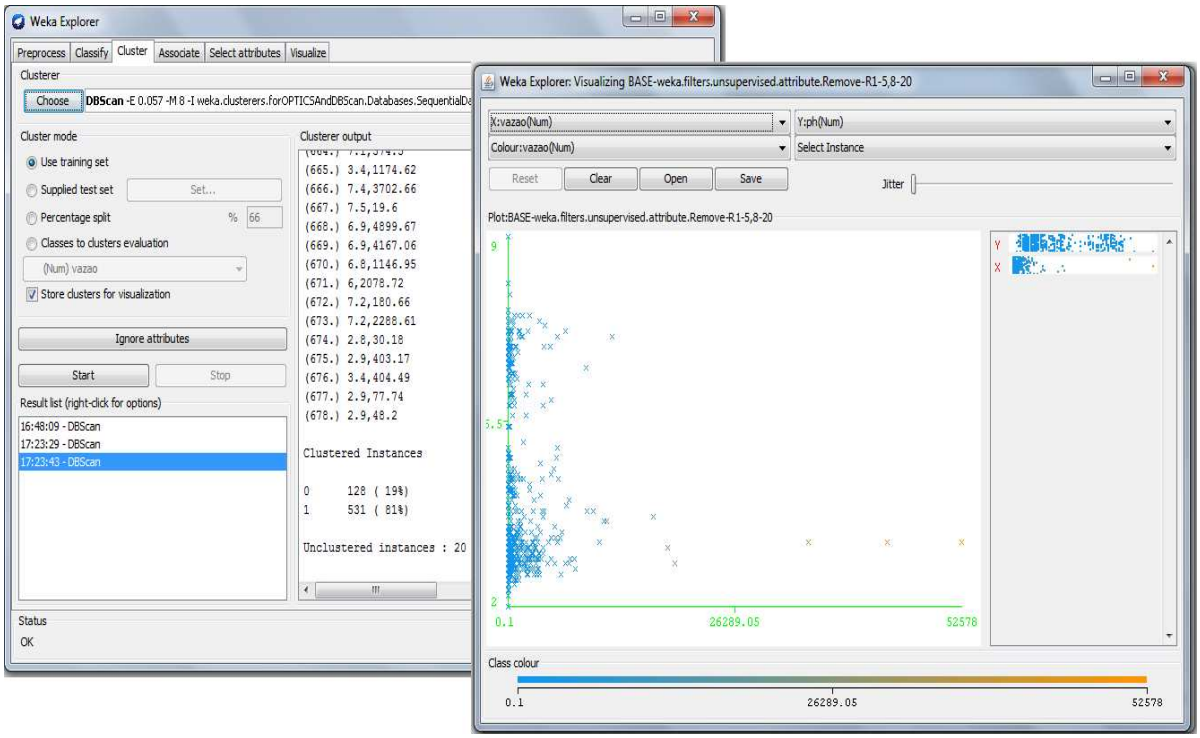

Figura 32. Resultados obtidos pelo DBSCAN na *Weka*

Observando-se que os resultados encontrados pela *Shell Orion* foram idênticos aos da ferramenta *Weka*, pode-se chegar à conclusão que em termos de resultados obtidos pelo DBSCAN, a *Shell Orion* implementa de maneira correta o algoritmo.

Na *Weka* os resultados gerados podem ser visualizados de maneira gráfica assim como na *Shell Orion*. Porém na *Orion* existe a possibilidade de os resultados do resumo serem agrupados de acordo com o atributo de saída desejado, enquanto na *Weka* 3.6 não existe essa possibilidade.

O tempo de processamento necessário para o algoritmo concluir a sua tarefa nas duas ferramentas também foi verificado. Para isso, utilizou-se uma determinada carga de dados, sem considerar os parâmetros do algoritmo e nem os atributos envolvidos na clusterização.

A medida de distância utilizada foi a euclidiana normalizada, os parâmetros η e ε foram fixados respectivamente em 4 e 10, a quantidade de atributos foi fixada em 2. A carga de dados foi definida replicando-se a base de dados, utilizando para isso o modelo de crescimento geométrico com 8 iterações, conforme descrito na secção 6.3.3.

Os resultados obtidos e a comparação entre os tempos de processamento da *Shell Orion* e *Weka* podem ser verificados na Tabela 20.

Tabela 20. Comparação de desempenho entre *Shell Orion* e *Weka* 3.6

Quantidade de registros	**Quantidade de atributos**	**Tempo de processamento (*Shell Orion*)**	**Tempo de processamento (*Weka* 3.6)**
679	2	00m: 00s: 106ms	00m: 00s: 070ms
1358	2	00m: 00s: 335ms	00m: 00s: 290ms
2716	2	00m: 01s: 188ms	00m: 01s: 160ms
5432	2	00m: 04s: 454ms	00m: 04s: 490ms
10864	2	00m: 18s: 925ms	00m: 18s: 870ms
21728	2	01m: 16s: 311ms	01m: 22s: 240ms
43456	2	05m: 43s: 463ms	06m: 10s: 090ms
86912	2	21m: 37s: 672ms	23m: 43s: 280ms

Pode-se chegar à conclusão que ambas as ferramentas possuem desempenho semelhante com relação ao parâmetro tempo de processamento. Na Tabela 20 se percebe que

quando a quantidade de registros aumenta, a *Shell Orion* tende a necessitar de um menor tempo de processamento do que a *Weka*.

6.3.4.1 Comparação entre as Médias de Tempo de Processamento do DBSCAN

Para verificação da média de tempo de processamento necessário ao DBSCAN para realizar a tarefa de clusterização, tanto o algoritmo na desenvolvido na *Shell Orion*, quanto o algoritmo disponível na *Weka* foram executados um número fixo de vezes com os mesmos parâmetros de entrada.

A carga de dados foi definida replicando-se a base de dados de monitoramento de indicadores ambientais, utilizando-se para isso o modelo de crescimento geométrico com 4 iterações (5432 registros). O algoritmo DBSCAN foi executado 400 vezes cada ferramenta, totalizando 800 observações.

Em cada execução, o tempo de processamento do algoritmo foi coletado. A análise dos dados foi realizada com o auxílio do pacote estatístico *Statistical Package for Social Sciences*[27] (SPSS).

Inicialmente os dados coletados foram tabulados e organizados a fim de permitir a sua análise. Utilizou-se o gráfico *boxplot* para identificação de possíveis *outliers* entre as observações, pois nesse tipo de gráfico é possível verificar claramente os valores atípicos no conjunto de observações (FIELD, 2009; OLIVEIRA, 2008).

Dentre as 800 observações, com a ajuda do gráfico *bloxpot* foi possível identificar 17 registros discrepantes do conjunto, sendo que desses, 10 registros faziam referência a *Weka* e sete são da *Shell Orion.* Esses valores estavam alterando a média e a variabilidade do conjunto de dados coletado, e portanto eles foram eliminados da análise.

[27] O SPSS para *Windows* oferece diversas possibilidades de cálculo estatístico e análise exploratória de dados. O *download* de uma versão de testes pode ser obtida em: (http://www-01.ibm.com/software/analytics/spss/)

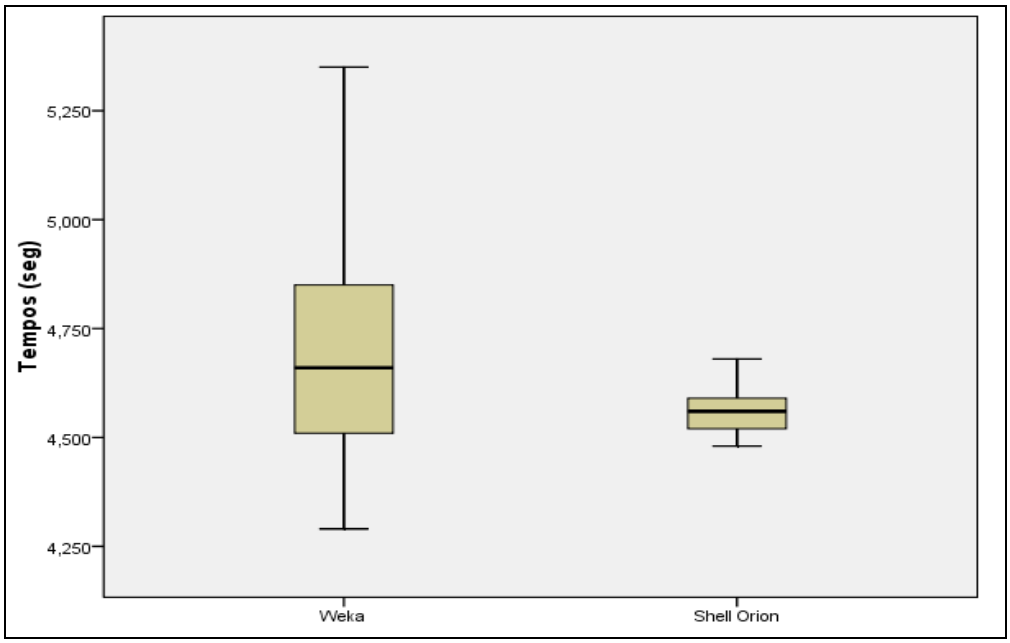

Figura 33. *Boxplot* após a exclusão dos *outliers*

Analisando o gráfico *boxplot* (Figura 33) pode ser percebido que enquanto os tempos de processamento da *Shell Orion* obtiveram boa estabilidade e apresentaram uma variância pequena, os dados obtidos da ferramenta *Weka* tiveram uma variância maior. A dispersão e a variabilidade maiores dos tempos de processamento da *Weka* podem ser atestadas pela amplitude, a variância e o desvio padrão apresentados para as tomadas de tempo obtidas nessa ferramenta (Tabela 21).

Tabela 21. Medidas de dispersão

Ferramenta	Mínimo	Máximo	Amplitude	Variância	Desvio Padrão
Weka 3.6	00m: 04s: 290ms	00m: 05s: 350ms	1.060	0.058	0.239867
Shell Orion	00m: 04s: 480ms	00m: 04s: 680ms	0.200	0.002	0.042670

Após a exclusão dos *outliers*, os dados foram submetidos ao teste de normalidade de *Kolmogorov-Smirnov*. Esse teste indicou a ausência de normalidade na distribuição de valores, obtendo um *p-value* inferior a 0.05 para ambas as ferramentas. A reprovação no teste de normalidade torna proibitivo o uso de testes paramétricos[28] tais como

[28] Testes paramétricos se caracterizam por suporem certa distribuição de probabilidades para a variável resposta (BARBETTA; REIS; BORNIA, 2010).

o teste *T* de *Student*[29], pois esse tipo de teste se baseia em uma distribuição normal de valores (FIELD, 2009).

Dessa maneira, optou-se por utilizar um teste não-paramétrico[30], visto que as suposições para aplicações de testes desse tipo são menos rígidas, o que possibilita uma aplicação mais generalizada. O teste não paramétrico de *U* de *Mann-Whitney* foi escolhido para comparação das médias de tempo de processamento. Trabalhou-se com as seguintes hipóteses:

$$H_0 : \mu_a = \mu_b \text{ e } H_1 : \mu_a \neq \mu_b$$

Onde:

a) $H_{0:}$ hipótese nula. Não existem diferenças entre os tempos médios de processamento;

b) $H_{1:}$ hipótese alternativa. Existem diferenças significativas entre as médias de tempo de processamento das duas ferramentas;

c) $\mu_{a:}$ média dos tempos de processamento da *Shell Orion*;

d) $\mu_{b:}$ média dos tempos de processamento da *Weka.*

O teste resultou em um *p-value* menor que o nível de significância pré-estabelecido (5%), portanto, rejeita-se a hipótese nula e se aceita a hipótese alternativa que diz que as médias de tempo de processamento são estatisticamente diferentes.

Como o resultado apresentado pelo teste *U* de *Mann-Whitney* mostrou evidência estatística que os tempos médios de processamento são diferentes, então é possível afirmar ao nível de significância de 5%, 95% de confiança, que os tempos de processamento do algoritmo DBSCAN não são iguais nas duas ferramentas.

---

[29] O teste *T* de *Student* é utilizado para comparar dois conjuntos de dados quantitativos, em termos de seus valores médios. Esse teste exige a hipótese de normalidade da população em questão (BARBETTA; REIS; BORNIA, 2010).

[30] Testes não-paramétricos são aplicados sem a necessidade de se fazer qualquer tipo de suposição sobre as distribuições e a origem das variáveis que estão sendo estudadas (BISQUERRA; SARRIERA; MARTINEZ, 2004).

Pelas evidências estatísticas encontradas, tais como a média (00m: 04s: 701ms para a *Weka* e 00m: 04s: 558ms para a *Shell Orion*) e a mediana (00m: 04s: 660ms para a *Weka* e 00m: 04s: 560ms para a *Orion*) e pelo resultado do teste *U* de *Mann-Whitney* (Tabela 22), é possível afirmar que o algoritmo DBSCAN apresentou menor tempo de processamento na *Shell Orion Data Mining Engine.*

Tabela 22. Teste *U* de *Mann-Whitney*

Ferramenta	**Média (seg)**	**Mínimo (seg)**	**Máximo (seg)**	***p-value***
Weka 3.6	00m: 04s. 701ms	00m: 04s. 290ms	00m: 05s. 350ms	0.000
Shell Orion	00m: 04s. 558ms	00m: 04s. 480ms	00m: 04s. 680ms	

Considerando todos os testes realizados, pode-se chegar à conclusão de que o algoritmo implementado obteve resultados satisfatórios em todos os aspectos, o que confirma a correta compreensão e implementação do algoritmo DBSCAN na *Shell Orion Data Mining Engine.*

CONCLUSÃO

A constante evolução das tecnologias de aquisição e armazenamento digital de dados possibilita a formação de repositórios de dados cada vez maiores. Nesse contexto o *data mining* possui fundamental importância na tarefa de extração de novos conhecimentos desses repositórios, auxiliando na tomada de decisão e na confirmação de conhecimentos já existentes.

Essa pesquisa fundamentou-se no entendimento dos principais conceitos relacionados à tarefa de clusterização, utilizando o método de densidade, por meio do algoritmo DBSCAN, que forma agrupamentos procurando por regiões com alta densidade de pontos no espaço de dados, possibilitando a detecção de *outliers* e a formação de *clusters* com formatos arbitrários.

Dentre as diversas dificuldades encontradas durante a realização da pesquisa, podem ser destacadas a escolha de índices adequados para avaliação de qualidade dos *clusters* gerados pelo algoritmo DBSCAN, a seleção de uma base de dados adequada e relevante para realizar a clusterização por meio do algoritmo desenvolvido e o entendimento dos testes estatísticos utilizados para avaliar o tempo de processamento do algoritmo implementado na *Shell Orion*. Os aspectos referentes ao fluxo de funcionamento do DBSCAN também representaram um obstáculo para a conclusão da pesquisa, porém, essa dificuldade foi superada com o auxílio da modelagem matemática do algoritmo.

Mesmo com as adversidades encontradas no decorrer da pesquisa, os objetivos foram atingidos. O algoritmo proposto foi disponibilizado na *Shell Orion*, inclusive com uma heurística que auxilia na definição dos parâmetros de entrada, visto que os resultados gerados pelo algoritmo são bastante afetados pela escolha destes. O módulo desenvolvido foi testado e a qualidade das partições geradas pelo algoritmo, utilizando as medidas de

distância implementadas, foram analisadas em conjunto com os índices de validação, gerando resultados satisfatórios e indicando o correto funcionamento do algoritmo implementado.

O tempo de processamento do módulo também foi avaliado. Pelos resultados obtidos, pode-se concluir que o tempo de processamento necessário ao algoritmo possui relação direta com o tamanho da base de dados utilizada, possuindo um crescimento de tempo aproximadamente linear com relação à quantidade de registros. A dimensionalidade também afeta consideravelmente o tempo de processamento do DBSCAN, comprovando que os atributos de entrada devem ser escolhidos com atenção.

Empregaram-se bases de dados com grandes quantidades de registros e alta dimensionalidade durante os testes com a finalidade de verificar o desempenho do algoritmo. Nesses casos, dispendeu-se bastante memória, pois para cada ponto deve ser armazenada a informação de quais pontos estão em sua vizinhança e qual a sua classificação atual.

Na comparação realizada com a ferramenta *Weka*, pode-se comprovar que os resultados obtidos pela pesquisa foram satisfatórios tanto em relação aos resultados encontrados, quanto em relação ao tempo de processamento, que foram semelhantes nas duas ferramentas. Os problemas que o algoritmo encontra com relação ao tempo de processamento e a excessiva utilização de memória verificados na *Shell Orion* também foram encontradas na *Weka*, quando realizadas grandes cargas de dados, concluindo que esse problema é inerente ao próprio algoritmo.

Também observou-se que o algoritmo implementado na *Shell Orion* obteve melhor média de tempo de processamento do que o DBSCAN disponibilizado na *Weka.* A comparação com uma ferramenta conhecida e bastante utilizada comprovou que o algoritmo implementado durante a pesquisa possui desempenho satisfatório, tanto de qualidade dos resultados gerados, quanto de desempenho.

Ao final, são descritas algumas sugestões de trabalhos futuros, visando dar continuidade ao desenvolvimento do projeto da *Shell Orion Data Mining Engine*:

a) adaptar o algoritmo DBSCAN para utilizar outras medidas de distância, como por exemplo, *Mahalanobis*, que implementa uma matriz de covariância entre os atributos, com a finalidade de calcular as relações existentes entre as propriedades, possibilitando maior flexibilidade na geração de *clusters*;
b) aplicar os resultados obtidos pelo algoritmo DBSCAN em outras tarefas de *data mining*, inclusive a análise de *outliers*, com o objetivo de descobrir padrões interessantes em conjuntos de dados classificados como *outliers* pelo DBSCAN;
c) pesquisar estruturas de dados que possam dar suporte de maneira mais eficiente ao processo de consulta de vizinhança realizada pelo algoritmo DBSCAN;
d) desenvolver outros algoritmos que utilizam o conceito de densidade, tais como o DENCLUE e o SNN;
e) disponibilizar na *Shell Orion* métodos de pré-processamento de dados, tais como técnicas de redução de atributos e de elementos.

REFERÊNCIAS

AGRAWAL, Rakesh et al. Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 1998, Seattle. **Proceedings…** . Seattle: ACM Press, 1998. p. 94-105.

ALEXANDRE, Nadja Zim; KREBS, Antonio Silvio Jornada. Discussão da aplicação do método do IQA na avaliação da qualidade das águas da região carbonífera de Santa Catarina. **Revista Tecnologia e Ambiente**, Criciúma, SC, v. 2, n. 1, p. 31-52, jun. 1996.

ALEXANDRE, Nadja Zim; KREBS, Antonio Silvio Jornada; VIERO, Ana Claudia. Qualidade das águas superficiais do município de Criciúma, SC - dados preliminares. **Revista Tecnologia e Ambiente**, Criciúma, SC, v. 1, n. 1, p. 29-54**,** jul. 1995.

ANKERST, Mihael; BREUNIG, Markus M.; KRIEGEL, Hans-Peter; SANDER, Jörg. OPTICS: ordering points to identify the clustering structure. In: ACM SIGMOD INTERNATIONAL CONFERENCE ON MANAGEMENT OF DATA, 1998, Philadelphia. **Proceedings…** . Philadelphia: ACM Press, 1999. p. 49-60.

APPEL, Ana Paula. **Métodos para o pré-processamento e mineração de grandes volumes de dados multidimensionais e redes complexas.** 2010. 179f. Tese (Doutorado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2010. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-13072010-101429/> Acesso em: 2011-04-18.

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. **Estatística**: para cursos de engenharia e informática. 3. ed. São Paulo: Atlas, 2010.

BATISTA, Gustavo Enrique de Almeida Prado Alves. **Pré-processamento de dados em aprendizado de máquina supervisionado.** 2003. 231f. Tese (Doutorado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2003. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-06102003-160219/> Acesso em: 2011-05-18.

BERRY, Michael J.; LINOFF, Gordon. **Data mining techniques:** for marketing, sales, and customer relationship management. 2. ed. Indianapolis: Wiley Publishing, 2004.

BEZDEK, James et al. **Fuzzy models and algorithms for pattern recognition and image processing**. New York: Springer, 2005.

BIGUS, Joseph P. **Data mining with neural networks:** solving business problems from application development to decision support**.** Hightstown: McGraw-Hill, 1996.

BISQUERRA, Rafael; SARRIERA, Jorge Castellá; MARTÍNEZ, Francesc. **Introdução a estatística**: enfoque informático com o pacote estatístico SPSS. Porto Alegre: Artmed, 2004.

BORTOLOTTO, Leandro Sehnem. **O Método de Redes Neurais pelo Algoritmo Kohonen para Clusterização na Shell Orion Data Mining Engine**. 2007. Trabalho de Conclusão de Curso - Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2007.

BOTELHO, Glenda Michele. **Seleção de características apoiada por mineração visual de dados.** 2011. 84f. Dissertação (Mestrado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2011. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-29032011-145542/> Acesso em: 2011-05-23.

BRAGA, Antônio de Pádua; DE CARVALHO, André Carlos Ponce de Leon Ferreira; LUDERMIR, Teresa Bernarda. **Redes neurais artificiais**: teoria e aplicações. Rio de Janeiro: LTC, 2000.

CABENA, Peter et al. **Discovering data mining:** from concept to implementation**.** New Jersey: Prentice Hall, 1998.

CARLANTONIO, Lando Mendonça di. **Novas metodologias para clusterização de dados**. 2001. 148p. Dissertação (Mestrado) - Departamento de Engenharia Civil, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2001. Disponível em: <http://wwwp.coc.ufrj.br.teses/mestrado/inter/2002/teses/di%20CARLANTONIO_LM_02_t_M_int.pdf/> Acesso em: 2011-03-21.

CARVALHO, Luís Alfredo Vidal de. **Data mining**: a mineração de dados no marketing, medicina, economia, engenharia e administração. Rio de Janeiro: Ciência Moderna, 2005.

CASAGRANDE, Diego Paz. **O Módulo da Técnica de Associação pelo Algoritmo Apriori no desenvolvimento da Shell de Data Mining Orion**. 2005. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2005.

CASSETTARI JUNIOR, José Márcio. **O Método de Lógica Fuzzy pelo Algoritmo Gustafson-Kessel na Tarefa de Clusterização da Shell Orion Data Mining Engine.** 2008. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina. 2008.

CASTILHO, Enrique; GUTIÉRREZ, José Manuel; HADI, Ali S. **Sistemas Expertos y Modelos de Redes Probabilísticas**. Madrid: Academia Española de Ingeniería, 1998.

CELEBI, Emri; ASLANDOGAN, Alp; BERGSTRESSER, Paul. Mining biomedical images with density-based clustering. In: INTERNATIONAL CONFERENCE ON INFORMATION TECHNOLOGY: CODING AND COMPUTING (ITCC'05), Washington. **Proceedings…** . Washington: IEEE Computer Society, 2005. p. 163-168.

CORDEIRO, João Paulo da Costa. **Extração de elementos relevantes em texto/páginas da world wide web**. 2003. 173f. Dissertação (Mestrado em Inteligência Artificial e Computação) - Faculdade de Ciências da Universidade do Porto. Universidade do Porto, Porto, 2003. Disponível em: <http://www.di.ubi.pt~jpaulo/publications/MSc-JPC.pdf> Acesso em: 2011-05-22.

CROTTI JUNIOR, Ademar. **O Método de Lógica Fuzzy pelos algoritmos Robust C-Prototypes e Unsupervised Robust C-Prototypes para a Tarefa de Clusterização na Shell Orion Data Mining Engine.** 2010. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina. 2010.

DASU, Tamraparni, JOHNSON, Theodore**. Exploratory data mining and data cleaning**. New Jersey: Wiley-Interscience, 2003.

ERMAN, Jeffrey; ARLITT, Martin; MAHANTI, Anirban. Traffic classification using clustering algorithms. In: SIGCOMM 2006 WORKSHOP ON MINING NETWORK DATA (MINENET 2006), 2006, New York. **Proceedings…** . New York: ACM Press, 2006. p. 281-286.

ERTÖZ, Levent; STEINBACH, Michael; KUMAR, Vipin. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In: SIAM INTERNATIONAL CONFERENCE ON DATA MINING, 2., 2003, Arlington. **Proceedings…** . Arlington: SIAM, 2003. p. 47-59.

ESTER, Martin; KRIEGEL, Hans-Peter; SANDER, Jörg; XU, Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING (KDD-96), 2., 1996, Portland. **Proceedings…** . Portland: AAAI Press, 1996 p. 226-231.

FACELLI, Katti. **Um framework para análise de agrupamento baseado na combinação multi-objetivo de algoritmos de agrupamento.** 2006. 183f. Tese (Doutorado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2006. Disponível em: <http://www. teses.usp.br/teses/disponiveis/55/55134/tde-12012007-082216/> Acesso em: 2011-05-21.

FAWCETT, Tom; PROVOST, Foster. Adaptative fraud detection. **Data mining and knowledge discovery**, Boston, p.291-316. set. 1997.

FAYYAD, Usama M.; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases**. AI Magazine**, Providence, v.17, n. 3, p. 37-54, autumn 1996.

FIELD, Andy. **Descobrindo a estatística usando o SPSS.** 2. ed. Porto Alegre: Artmed, 2009.

FRIETMAN, E.; HILL, M.; KHOE, G. A kohonen neural network controlled all-optical router system. **International journal of computer research**, Sofia, v. 10, n. 2, p.251-267. 2001**.**

FURLAN, José Davi. **Modelagem de objetos através da UML**. São Paulo: Makron Books: 1998.

GALATTO, Sérgio Luciano et al. Emprego de coberturas secas no controle da drenagem ácida de mina: estudos em campo. **Engenharia Sanitária e Ambiental,** Rio de Janeiro, v. 12, n. 2, p. 229-236, jun. 2007.

GALVANI, Emerson; LUCHIARRI, Ailton. Critérios para classificação de anos com regime pluviométrico normal, seco e úmido. In: SIMPÓSIO BRASILEIRO DE CLIMATOLOGIA GEOGRÁFICA, 6., 2004, Aracaju. **Anais...** . Aracaju: Associação Brasileira de Climatologia, 2004. v. 1, p. 20-30.

GAN, Goujan; MA, Chaoqun; WU, Jianhong. **Data clustering**: theory, algorithms and applications. Philadelphia: SIAM, 2007.

GERSTING, Judith L. **Fundamentos matemáticos para a ciência da computação**. 3. ed. Rio de Janeiro: LTC, 1995.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel Lopes. **Data mining**: uma guia prático: conceitos, técnicas, ferramentas, orientações e aplicações. Rio de Janeiro: Elsevier, 2005.

GTA - Grupo Técnico de Assessoramento (DNPM, CPRM, SIECESC, FATMA, MPF). **Terceiro Relatório de Monitoramento dos Indicadores Ambientais**. Versão 03. Revisão Final. Criciúma, 2009.

GUEDES, Gilleanes Thorwald Araujo. **UML**: uma abordagem prática. 3. ed. São Paulo: Novatec, 2008.

GUNOPULOS, Dimitrios. Clustering Overview and Applications. In: LIU, Ling; ÖZSU, Tamer. **Encyclopedia of Database Systems**. New York: Springer, 2009. p. 383-387.

HAN, Jiawei. KAMBER, Micheline. **Data mining**: concepts and techniques. 2. ed. San Francisco: Morgan Kaufmann, 2006.

HAN, Jiawei; KAMBER, Micheline; TUNG, Anthony K. H. Spatial clustering methods in data mining: a survey**.** In: MILLER, Harvey J.; HAN, Jiawei. **Geographic Data Mining and Knowledge Discovery**: Research Monographs in GIS, London: Taylor and Francis, 2001. p. 201-230.

HARRISON, Thomas H. **Intranet data warehouse**. São Paulo*:* Berkeley, 1998.

HAYKIN, Simon. **Redes neurais:** princípios e técnicas. 2. ed. Porto Alegre: Bookman, 2000.

HINNEBURG, Alexander; KEIM, Daniel A. An efficient approach to clustering in large multimedia databases with noise**.** In: KNOWLEDGE DISCOVERY AND DATA MINING, 4., 1998, New York. **Proceedings…** . New York: AAAI Press, 1998. p. 58-65

HÖPPNER, Frank et al. **Fuzzy cluster analysis**: methods for classification, data analysis, and image recognition. Chichester: Wiley-Interscience, 1999.

JAIN, Anil K.; DUBES, Richard C. **Algorithms for clustering data.** Englewood Cliffs: Prentice Hall, 1988.

JAIN, Anil K.; MURTY, M. N.; FLYNN, P. J. Data clustering: a review. **ACM Computing Surveys (CSUR)**, New York, p.264-323. sep. 1999.

KAUFMAN, Leonard; ROUSSEEAW, Peter J. **Finding groups in data:** an introduction to cluster analysis. Hoboken: Wiley-Interscience, 1990.

KRIEGEL, Hans-Peter et al. Density-based clustering. In: PEDRYCZ, Witold (Org.). **Wiley interdisciplinary reviews**: data mining and knowledge discovery, New York, Wiley, 2011. p. 231-240.

LAROSE, Daniel T. **Discovering knowledge in data:** an introduction to data mining. Hoboken: Wiley-Interscience, 2005.

MARTENS, Harald; NAES, Tormod. **Multivariate calibration.** Chichester: **Wiley-Interscience,** 1993.

MARTINS, Denis Piazza. **O Algoritmo de Particionamento K-means na Tarefa de Clusterização da Shell Orion Data Mining Engine**. 2007. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2007.

METZ, Jean. **Interpretação de clusters gerados por algoritmos de clustering hierárquico.** 2006. 95f. Dissertação (Mestrado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2006. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-14092006-090701/> Acesso em: 2011-05-23.

MILLIGAN, Glenn W.; COOPER, Martha C. An examination of procedures for determining the Number of clusters in a data set. **Psychometrika**, Colombus, p. 159-179. Jun. 1985.

MINGOTTI, Sueli Aparecida. **Análise de dados através de métodos de estatística multivariada**: uma abordagem aplicada. Belo Horizonte: Editora UFMG, 2005.

MONDARDO, Ricardo Lineburger. **O algoritmo C4.5 na tarefa de classificação da Shell Orion Data Mining Engine.** 2009. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina. 2009.

NALDI, Murilo Coelho. **Técnicas de combinação para agrupamento centralizado e distribuído de dados.** 2011. 245f. Tese (Doutorado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2011. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-16032011-113154/> Acesso em: 2011-05-20.

OLIVEIRA, Camillo Jorge Santos. **Classificação de imagens coletadas na web**. 2001. 73f. Dissertação (Mestrado) - Curso de Pós-Graduação em Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte, 2001. Disponível em: <http://laplace.dcc.ufmg.br/npdi/uploads/96a40bea-e18d-5936.pdf/> Acesso em: 2011-04-19.

OLIVEIRA, Tatyana Bitencourt Soares de. **Clusterização de dados utilizando técnicas de redes complexas e computação bioinspirada**. 2008. 95f. Dissertação (Mestrado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2008. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-01042008-142253/> Acesso em: 2011-05-23.

PAL, Sankar K.; MITRA, Pabitra. **Pattern recognition algorithms for data mining**: scalability, knowledge discovery and soft granular computing. Florida: Chapman & Hall, 2004.

PATERLINI, Adriano Arantes. I**mersão de espaços métricos em espaços multidimensionais para indexação de dados usando detecção de agrupamentos.** 2011. 90f. Dissertação (Mestrado em Ciências da Computação e Matemática Computacional)- Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo, São Carlos, 2011. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-25042011-155810/> Acesso em: 2011-04-20.

PAVEI, Paula Tramontim. **Caracterização e estudo do comportamento de hidrocarbonetos policíclicos aromáticos em ecossistemas aquáticos contaminados pelas atividades mineração de carvão**. 2007. 110f. Dissertação (Mestrado) - Programa de Pós-Graduação em Ciências Ambientais, Universidade do Extremo Sul Catarinense, Criciúma, 2007. Disponível em: <http://www.bib.unesc.net/biblioteca/sumario/000035/0000359F.pdf/> Acesso em: 2011-09-10.

PELEGRIN, Diana Colombo. **A Tarefa de Classificação e o Algoritmo ID3 para Indução de Árvores de Decisão na Shell de Data Mining Orion**. 2005. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2005.

PEREGO, Daniel. **O Método de Lógica Fuzzy pelo algoritmo GATH-GEVA na tarefa de clusterização da Shell Orion Data Mining Engine.** 2009. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina. 2009.

PYLE, Dorian. **Data preparation for data mining**. San Francisco: Morgan Kaufmann, 1999.

RAIMUNDO, Lidiane Rosso. **O Algoritmo CART na Tarefa de Classificação da Shell Orion Data Mining Engine**. 2007. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina, 2007.

SASSI, Renato José. **Uma arquitetura híbrida para descoberta de conhecimento em bases de dados:** teoria dos rough sets e redes neurais artificiais mapas auto organizáveis. 2006. 169 f. Tese (Doutorado em Sistemas Eletrônicos) – Escola Politécnica, Universidade de São Paulo. São Paulo, 2006. Disponível em: <http://www.teses.usp.br/teses/disponiveis/3/3142/tde-16032007-163930/> Acesso em: 2011-06-15.

SCOTTI, Ana Paula. **O Método de Redes Neurais com Função de Ativação de Base Radial para a Tarefa de Classificação na Shell Orion Data Mining Engine**. 2010. Trabalho de Conclusão de Curso – Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, Santa Catarina. 2010.

SFERRA Heloisa Helena; CORRÊA, Ângela M. C. Jorge. Conceitos e aplicações de data mining**. Revista de Ciência e Tecnologia**, Piracicaba, v.11, n. 22, p. 19-34, jul. 2003.

SHALABI, Luai Al, SHAABAN, Zyad, KASASBEH, Basel, Data mining: a preprocessing engine**, Journal of Computer Science**, Amman, p. 735-739. Jun. 2006.

SIVANANDAM, S. N.; SUMATHI, S. **Introduction to data mining and its applications.** Berlim: Springer, 2006.

SOUTO, Rodrigo Fontes. **Estimadores robustos para sistemas lineares e não-lineares**. 2009. 98f. Dissertação (Mestrado) **–** Departamento de Engenharia Elétrica. Universidade de Brasília, Brasília, 2009. Disponível em: <http://repositorio.bce.unb.br/handle/10482/4300/> Acesso em: 2011-09-15.

WITTEN, Ian H.; FRANK, Eibe; HALL, Mark A. **Data mining practical machine learning tools and techniques**. 3. ed. Burlington: Morgan Kaufmann, 2011.

WRÓBLEWSKA, Anna et al. Two stage detection and clustering of microcalcifications in mammograms**.** In: INTERNATIONAL CONFERENCE OF MEDICAL PHYSICS (ICMP 2005), 14., 2005, Nuremberg. **Proceedings…** .Nuremberg: IOMP, 2005. p. 56-57.

XU, R., WUNSCH, D.. Survey of clustering algorithms, **IEEE Transactions on Neural Networks**, New York, v.16, n. 3, p. 645-678, May 2005.

ZHOU, Zhi-Hua. Three Perspectives of data mining. **Artificial Inteligence**, London, p. 139-146. Jan. 2003

APÊNDICE A - *SHELL ORION DATA MINING ENGINE*

Atualmente a *Shell Orion* está organizada em módulos diferentes para cada tarefa de DM, possuindo métodos que contemplam as tarefas de classificação, clusterização e associação. Cada algoritmo implementado é independente dos demais, necessitando de informações específicas para seu funcionamento adequado de acordo com a tarefa que se deseja realizar.

Para a tarefa de associação encontra-se disponível o algoritmo *Apriori*, um método tradicional para a descoberta de regras de associação. Desenvolvido na *Shell Orion* pelo acadêmico Diego Paz Casagrande, no ano de 2005 esse algoritmo gera regras de associação baseado nos valores de suporte[31] mínimo e de confiança[32] mínima dos *itemsets*[33]·. Nó módulo de classificação estão disponíveis os algoritmos CART, ID3, C4.5 e RBF.

O algoritmo ID3, proposto por Ross Quinlan em 1979, foi disponibilizado na *Shell Orion* no ano de 2005 pela acadêmica Diana Colombo Pelegrin, sendo um dos primeiros algoritmos incorporados a ferramenta. O ID3 trabalha com atributos nominais e utiliza o critério de ganho de informação[34] para a construção das árvores de decisão (OLIVEIRA, 2001). A abordagem adotada pelo algoritmo para a construção de árvores de decisão é de cima para baixo (*top-down*), recursivamente, pois ao escolher um atributo para um nó, partindo da raiz, é aplicado o mesmo algoritmo aos descendentes desse nó, até que

[31] O número de vezes em que a união de conjuntos de itens (X U Y) ocorre deve ser superior ou igual ao valor de suporte mínimo em relação ao total de transações da base de dados,ou seja,busca medir a freqüência de ocorrência de uma associação,afim de identificar associações com uma quantidade expressiva de ocorrências (GOLDSCHIMIDT; PASSOS, 2005).

[32] O número de transações na base de dados em que a união de conjuntos de itens (X U Y) ocorrer em relação ao número de vezes em que X ocorrer deve ser igual ou superior a um valor de confiança mínima estabelecido (GOLDSCHIMIDT; PASSOS, 2005).

[33] Uma regra de associação é uma implicação da forma $X \rightarrow Y$, onde X e Y são conjuntos de itens (GOLDSCHMIDT; PASSOS,2005).

[34] O ganho de informação é uma medida estatística utilizada para a construção das árvores de decisão e segundo Pelegrin (2005) o emprego desse conceito no algoritmo ID3 possibilita minimizar a profundidade final da árvore de decisão.

certos critérios de parada sejam atingidos e a execução é encerrada (CORDEIRO, 2003).

Disponibilizado em 2007, pela acadêmica Lidiane Rosso Raimundo, o CART foi desenvolvido em 1984 pelos estatísticos Leo Breiman, Jerome Friedman, Richard Oslen e Charles Stone, sendo um dos principais algoritmos baseados em árvores de decisão[35] para a tarefa de classificação. As árvores de decisão produzidas pelo CART são estritamente binárias, contendo sempre dois subconjuntos para cada nó de decisão e geradas recursivamente (LAROSE, 2005).

O algoritmo C4.5, desenvolvido por Ross Quinlan, no ano de 1987, foi o tema do trabalho de conclusão de curso do acadêmico Ricardo Lineburger Mondardo no ano de 2009. Assim como o CART, o C4.5 procura abstrair árvores de decisão a partir de uma abordagem recursiva de particionamento. Porém ao contrario do CART, o C4.5 não se restringe a geração de árvores binárias, produzindo árvores com formatos variados e pode trabalhar com atributos nominais e numéricos (LAROSE,2005).

A acadêmica Ana Paula Scotti acrescentou ano de 2010 ao módulo de classificação da *Orion* o método de redes neurais com função de ativação de base radial (RBF). Redes neurais são estruturas nas quais os neurônios estão dispostos em camadas e interligados por conexões conhecidas como pesos sinápticos que representam o conhecimento da rede (SCOTTI, 2010). Estas estruturas possuem a capacidade de classificar padrões desconhecidos adequando-se à resolução de problemas onde se tem pouco conhecimento das relações entre atributos e classes (HAYKIN, 2000, tradução nossa).

O módulo de clusterização da *Shell Orion Data Mining Engine* disponibiliza os algoritmos K-*means*, *Kohonen*, *Robust C-Prototypes*, *Unsupervised Robust C-Prototypes*, *Fuzzy C-means*, Gath-Geva e Gustafson-Kessel, sendo o módulo que disponibiliza o maior número de opções de algoritmos para a realização da tarefa a que se destina.

[35] Modelo de conhecimento em que cada nó interno da árvore representa uma decisão sobre um atributo que determina como os dados estão particionados pelos seus nós filhos (GOLDSCHMIDT; PASSOS, 2005).

No ano de 2007 foi adicionado pelo acadêmico Dênis Piazza Martins o clássico algoritmo *K-means* para a tarefa de clusterização. No *K-means* primeiramente é informado o número *K* de *clusters* desejados, sendo que o algoritmo escolhe em seguida *K* pontos aleatórios como elementos centrais de cada *cluster*. Após isso, cada registro é atribuído, mediante cálculo de distância, ao *cluster* com centro mais próximo e é recalculado o centróide de cada *cluster*. Esse é um processo que continua iterativamente até que os pontos centrais de cada *cluster* se estabilizem e sejam sempre os mesmos (WITTEN; FRANK; HALL, 2011, tradução nossa)

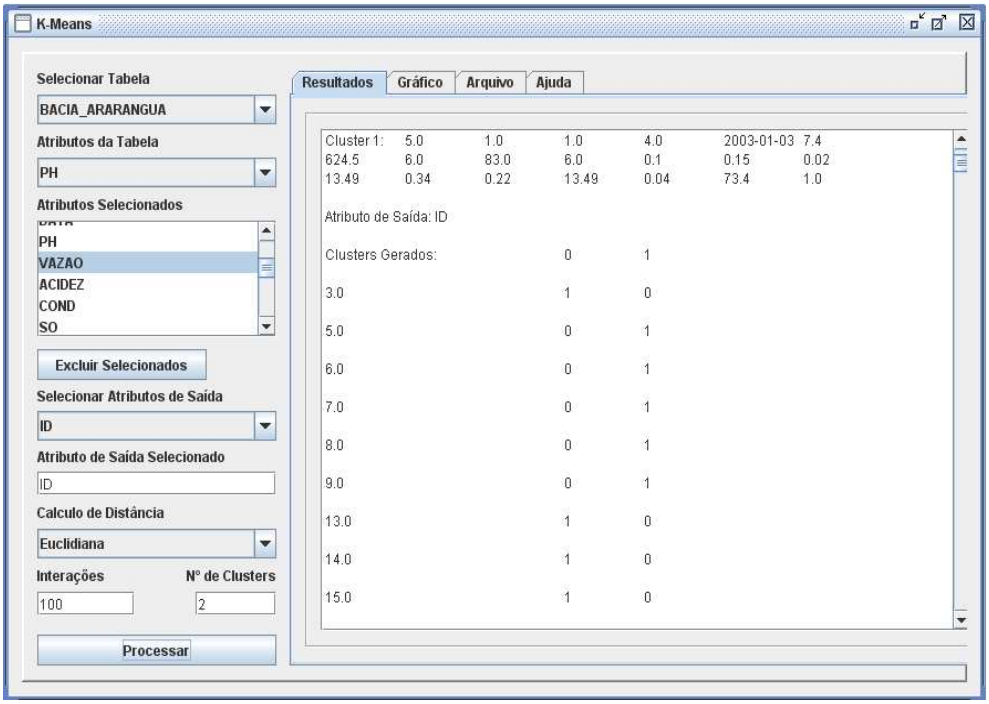

Figura 34. Algoritmo *K-means* na *Shell Orion*

Também em 2007, foi disponibilizado o método de redes neurais artificiais por meio do algoritmo de *Kohonen*, implementado pelo acadêmico Leandro Sehnem Bortolotto. Também conhecido por Mapa Auto-Organizável *(Self-Organizing Map* - SOM), o algoritmo de *Kohonen* é um tipo de rede neural artificial baseada em aprendizado não-supervisionado[36], sendo capaz de mapear um conjunto de dados, de um espaço de entrada

[36] No aprendizado não supervisionado, o conjunto de dados de entrada é composto por exemplos não rotulados, sendo que não é possível saber a qual classe pertence cada exemplo (METZ, 2006).

multidimensional, em um conjunto finito de neurônios organizados de forma unidimensional ou bidimensional (SASSI, 2006).

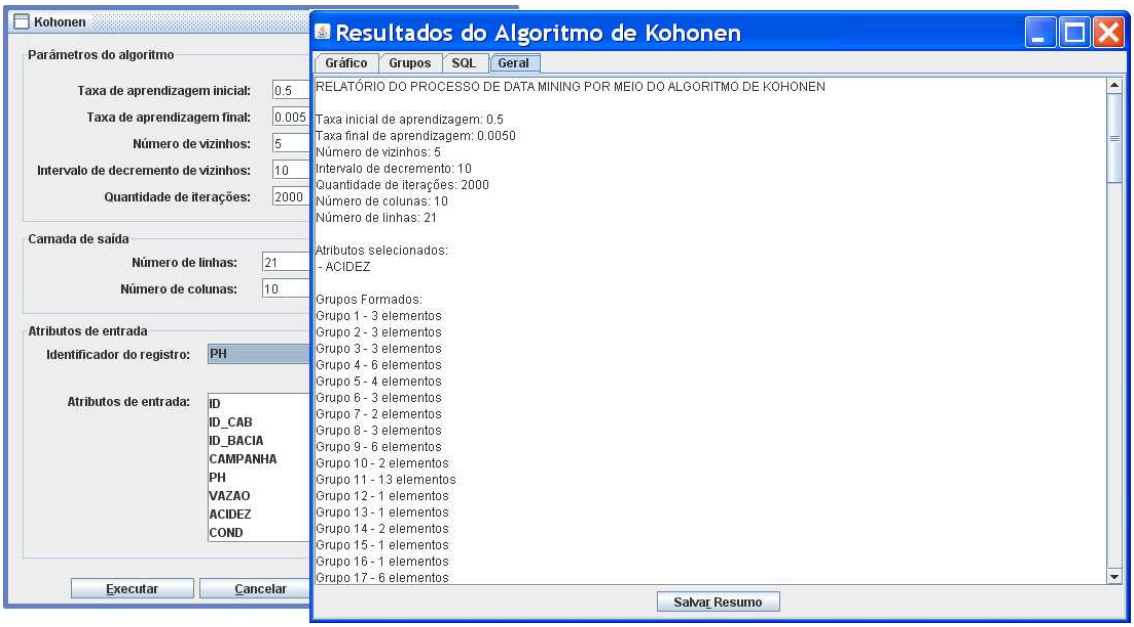

Figura 35. Algoritmo de *Kohonen* na *Shell Orion*

O método de lógica *fuzzy* foi introduzido na *Shell Orion em* 2008 por meio do algoritmo *Gustafson-Kessel* (GK) pelo acadêmico José Marcio Cassettari Júnior. Esse algoritmo foi proposto em 1979 por Donald E. Gustafson e Wiliam C. Kessel, e surgiu como uma extensão do algoritmo *Fuzzy C-Means* (FCM), sendo que o GK utiliza uma matriz de covariância[37] *fuzzy* para encontrar *cluster* de vários formatos geométricos, superando assim as limitações impostas pelo algoritmo FCM que tende a produzir *clusters* esféricos, devido à utilização da medida de distância euclidiana (HÖPPNER et al, 1999, tradução nossa).

[37] Matriz que contem em sua diagonal principal as variâncias das colunas, ou seja, o afastamento de seus elementos em torno da sua média, e nas posições restantes as covariâncias, que são como os elementos variam de acordo com os valores esperados (CROTTI JÚNIOR, 2010).

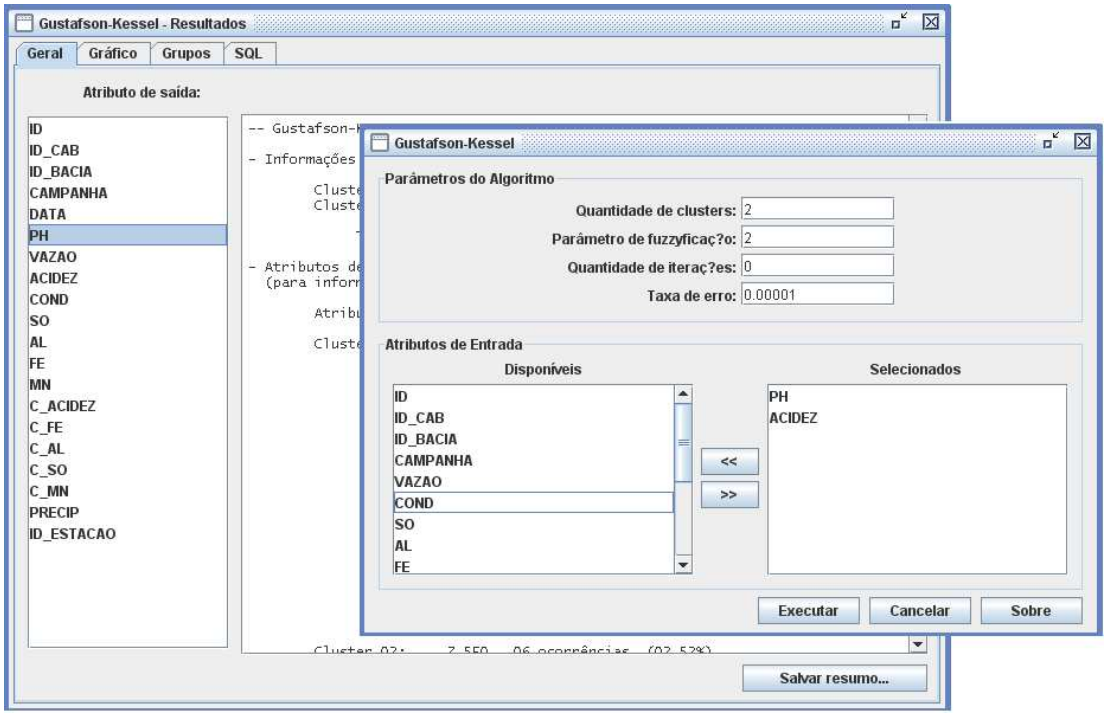

Figura 36. Algoritmo GK na *Shell Orion Data Mining Engine*

No ano de 2009, dando continuidade a expansão das funcionalidades da *Orion*, foi implementado pelo acadêmico Daniel Perego, o algoritmo *Gath-Geva*, que também utiliza lógica *fuzzy* no módulo de clusterização da ferramenta.

Considerado uma evolução do GK, o *Gath-Geva*, proposto por Isak Gath e Amir Geva, em 1989, utiliza uma matriz de covariância, assim como o GK, encontrando *cluster* com formatos geométricos variados, porém essa matriz é usada em conjunto com um termo de distância exponencial, contornando assim o problema que o algoritmo GK enfrenta em relação à geração de *cluster* com volumes muito diferentes (HÖPPNER et al, 1999, tradução nossa).

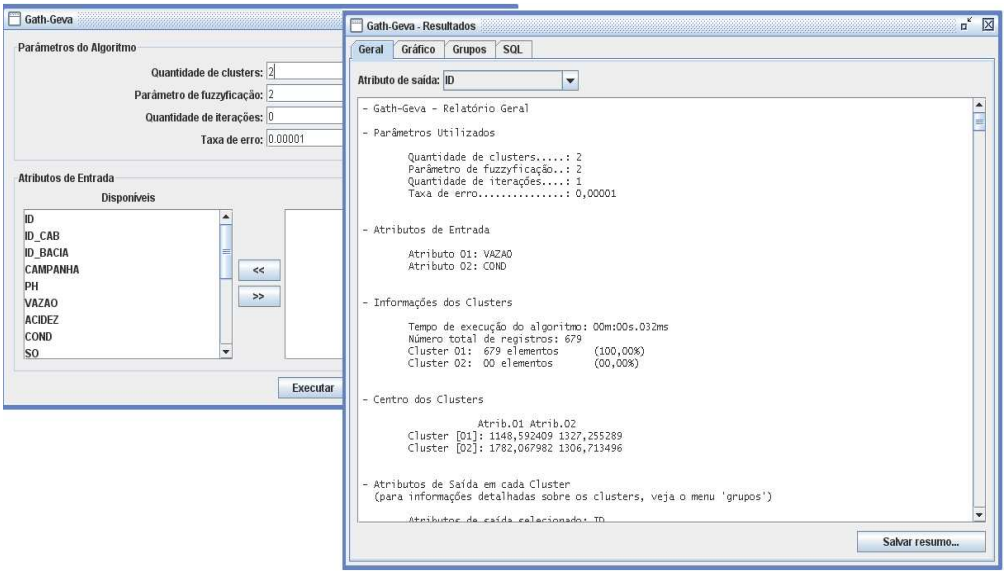

Figura 37. Algoritmo *Gath-Geva* na *Shell Orion Data Mining Engine*

No trabalho de conclusão de curso do de Ademar Crotti Júnior, no ano de 2010, foram disponibilizados os algoritmos *Robust C-Prototypes* (RCP), *Unsupervised Robust C-Prototypes* (URCP) e *Fuzzy C Means* (FCM) que utilizam lógica *fuzzy* para a tarefa de clusterização. O FCM foi proposto por James C. Bezdek em 1973, inspirado no algoritmo *K-means* e usa a distância euclidiana, gerando *clusters* esféricos, assim como o *K-means*, porém ao contrario deste, o FCM utiliza o elemento *fuzificador*, para determinar o grau de relação existente entre os elementos (BEZDEK et al,2005).

O RCP é um algoritmo inspirado no FCM e foi proposto por Frigui e Krishnapuram em 1996. Considerando as limitações impostas pelo FCM, que gera resultados insatisfatórios quando trabalha com um conjunto de dados contaminados por *outliers,* esses dois pesquisadores acrescentaram um estimador robusto[38] ao algoritmo FCM, visando assim minimizar os problemas causados por *outliers* na geração de *clusters*, definindo assim o algoritmo RCP (KRUSE et al, 2007, tradução nossa).

Um dos parâmetros de entrada para o algoritmo RCP é a quantidade de *clusters* que se deseja obter, porém não se trata de uma tarefa trivial a obtenção desse dado

[38] Tais estimadores são ditos robustos no sentido em que eles tentam limitar, de certa forma, os efeitos de incertezas, no desempenho médio do estimador (SOUTO, 2009).

antecipadamente. Então Frigui e Krishnapuram também propuseram uma extensão do RCP que pode ser usada quando essa informação não está disponível. Essa extensão é conhecida como URCP.

Como parâmetro de entrada do URCP, ao invés de se informar o número de *clusters* desejados, é informado o número máximo de *clusters* que podem ser gerados, sendo essa a principal diferença entre o RCP e o URCP.

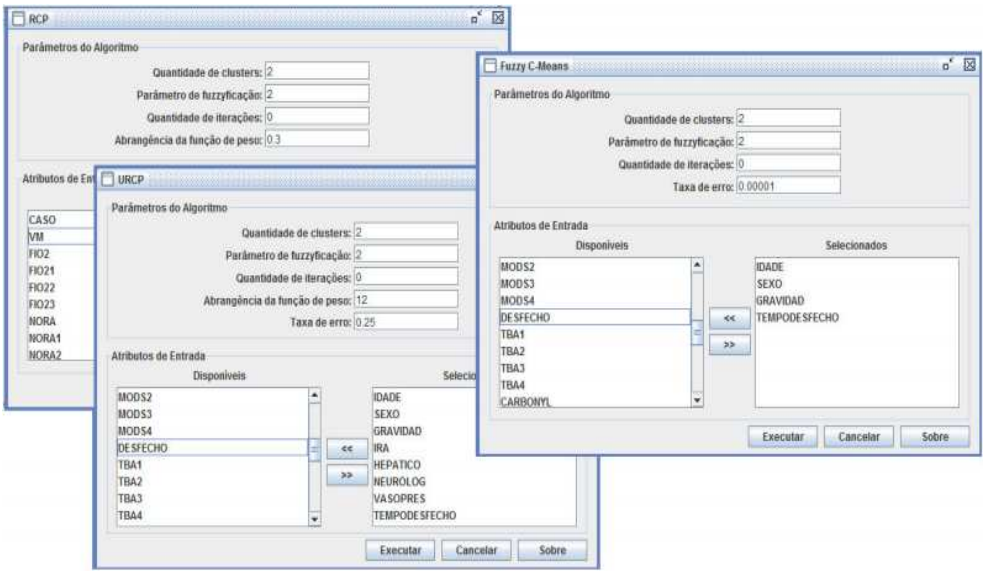

Figura 38. Algoritmos RCP, URCP e FCM na *Shell Orion*
Fonte: CROTTI JÚNIOR, A. (2010)

APÊNDICE B – MODELAGEM MATEMÁTICA UTILIZANDO A DISTÂNCIA *MANHATTAN*

Neste apêndice é demonstrada à modelagem matemática do algoritmo DBSCAN utilizando a distância *Manhattan*. Para demonstração foi utilizada uma base de dados contendo 7 elementos (x_1, x_2, x_3, x_4, x_5, x_6, x_7) com 2 atributos cada (a, b). A Tabela 23 exemplifica os valores utilizados:

Tabela 23. Base de dados utilizada na modelagem do algoritmo

Atributos	x_1	x_2	x_3	x_4	x_5	x_6	x_7
a	5,1	4,9	4,7	4,6	5,0	5,3	8,0
b	3,5	3,0	3,2	3,1	3,6	3,5	3,0

Para demonstração do funcionamento do algoritmo foram definidos os seguintes parâmetros de entrada:

c) **raio da ε-vizinhança de um ponto:** especifica o valor para a ε-vizinhança de um objeto. Foi definido o valor 0,3;

d) **número mínimo de pontos (η):** parâmetro que específica o número mínimo de pontos, no dado raio de ε-vizinhança, que um ponto precisa possuir para ser considerado um ponto central. Definiu-se o valor 3.

O primeiro passo consiste em montar a matriz de dissimilaridade utilizando para isso a distância *Manhattan*:

$$d\left(x_{i,}x_{j}\right)=\sum_{l=1}^{d}\left|\left(x_{il}-x_{jl}\right)\right|$$

$$dist\left(x_1,x_2\right)=\left|5{,}1-4{,}9\right|+\left|3{,}5-3{,}0\right|=\left|0{,}2\right|+\left|0{,}5\right|=0{,}7$$

$$dist\left(x_1,x_3\right)=\left|5{,}1-4{,}7\right|+\left|3{,}5-3{,}2\right|=\left|0{,}4\right|+\left|0{,}3\right|=0{,}7$$

$$dist\left(x_1,x_4\right)=\left|5{,}1-4{,}6\right|+\left|3{,}5-3{,}1\right|=\left|0{,}5\right|+\left|0{,}4\right|=0{,}9$$

$$dist\left(x_1,x_5\right)=\left|5{,}1-5{,}0\right|+\left|3{,}5-3{,}6\right|=\left|0{,}1\right|+\left|-0{,}1\right|=0{,}2$$

$$dist\left(x_1,x_6\right)=\left|5{,}1-5{,}3\right|+\left|3{,}5-3{,}5\right|=\left|-0{,}2\right|+\left|0{,}0\right|=0{,}2$$

$$dist(x_1, x_7) = |5{,}1 - 8{,}0| + |3{,}5 - 3{,}0| = |-2{,}9| + |0{,}5| = 3{,}4$$

$$dist(x_2, x_3) = |4{,}9 - 4{,}7| + |3{,}0 - 3{,}2| = |0{,}2| + |-0{,}2| = 0{,}4$$

$$dist(x_2, x_4) = |4{,}9 - 4{,}6| + |3{,}0 - 3{,}1| = |0{,}3| + |-0{,}1| = 0{,}4$$

$$dist(x_2, x_5) = |4{,}9 - 5{,}0| + |3{,}0 - 3{,}6| = |-0{,}1| + |-0{,}6| = 0{,}7$$

$$dist(x_2, x_6) = |4{,}9 - 5{,}3| + |3{,}0 - 3{,}5| = |-0{,}4| + |-0{,}5| = 0{,}9$$

$$dist(x_2, x_7) = |4{,}9 - 8{,}0| + |3{,}0 - 3{,}0| = |-3{,}1| + |0{,}0| = 3{,}1$$

$$dist(x_3, x_4) = |4{,}7 - 4{,}6| + |3{,}2 - 3{,}1| = |0{,}1| + |0{,}1| = 0{,}2$$

$$dist(x_3, x_5) = |4{,}7 - 5{,}0| + |3{,}2 - 3{,}6| = |-0{,}3| + |-0{,}4| = 0{,}7$$

$$dist(x_3, x_6) = |4{,}7 - 5{,}3| + |3{,}2 - 3{,}5| = |-0{,}6| + |-0{,}3| = 0{,}9$$

$$dist(x_3, x_7) = |4{,}7 - 8{,}0| + |3{,}2 - 3{,}0| = |-3{,}3| + |0{,}2| = 3{,}5$$

$$dist(x_4, x_5) = |4{,}6 - 5{,}0| + |3{,}1 - 3{,}6| = |-0{,}4| + |-0{,}5| = 0{,}9$$

$$dist(x_4, x_6) = |4{,}6 - 5{,}3| + |3{,}1 - 3{,}5| = |-0{,}7| + |-0{,}4| = 1{,}1$$

$$dist(x_4, x_7) = |4{,}6 - 8{,}0| + |3{,}1 - 3{,}0| = |-3{,}4| + |0{,}1| = 3{,}5$$

$$dist(x_5, x_6) = |5{,}0 - 5{,}3| + |3{,}6 - 3{,}5| = |-0{,}3| + |0{,}1| = 0{,}4$$

$$dist(x_5, x_7) = |5{,}0 - 8{,}0| + |3{,}6 - 3{,}0| = |-3{,}0| + |0{,}6| = 3{,}6$$

$$dist(x_6, x_7) = |5{,}3 - 8{,}0| + |3{,}5 - 3{,}0| = |-2{,}7| + |0{,}5| = 3{,}2$$

Tendo as distâncias entre todos os pontos calculados, então é possível montar a matriz de dissimilaridade.

$$\begin{bmatrix} 0 & 0{,}7 & 0{,}7 & 0{,}9 & 0{,}2 & 0{,}2 & 3{,}4 \\ 0{,}7 & 0 & 0{,}4 & 0{,}4 & 0{,}7 & 0{,}9 & 3{,}1 \\ 0{,}7 & 0{,}4 & 0 & 0{,}2 & 0{,}7 & 0{,}9 & 3{,}5 \\ 0{,}9 & 0{,}4 & 0{,}2 & 0 & 0{,}9 & 1{,}1 & 3{,}5 \\ 0{,}2 & 0{,}7 & 0{,}7 & 0{,}9 & 0 & 0{,}4 & 3{,}6 \\ 0{,}2 & 0{,}9 & 0{,}9 & 1{,}1 & 0{,}4 & 0 & 3{,}2 \\ 3{,}4 & 3{,}1 & 3{,}5 & 3{,}5 & 3{,}6 & 3{,}2 & 0 \end{bmatrix}$$

O conjunto auxiliar contendo todos os pontos ainda não visitados pelo algoritmo inicialmente possui os seguintes elementos:

$$C_a = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$$

O algoritmo escolhe o primeiro ponto no conjunto C_a, ou seja, x_1. Então é verifica a $N_\varepsilon(x_1)$ para saber se a cardinalidade do ponto é igual ou excede η, ou seja, verificar a condição: $Card\ (N_\varepsilon(x_1)) \geq \eta$. Para isso, o algoritmo DBSCAN verifica a matriz de dissimilaridade a fim de saber quantos pontos a ε-vizinhança de x_1 contêm:

$$dist(x_1, x_2) = 0{,}7$$
$$dist\ (x_1, x_3) = 0{,}7$$
$$dist(x_1, x_4) = 0{,}9$$
$$dist\ (x_1, x_5) = 0{,}2$$
$$dist\ (x_1, x_6) = 0{,}2$$
$$dist\ (x_1, x_7) = 3{,}4$$

Consultando a $N_\varepsilon(x_1)$ o DBSCAN constatou que os pontos x_5 e x_6 estão na ε-vizinhança de x_1, visto que $dist(x_1, x_5) \leq \varepsilon$ e $dist(x_1, x_6) \leq \varepsilon$ para ε= 0,3 e, portanto são diretamente alcançáveis por densidade a partir de x_1. Isso implica que x_1 é um ponto central, pois a $N_\varepsilon(x_1) \geq \eta$ para η = 3.

Então um *cluster* C_1 é formado contendo o ponto central x_1 e todos os pontos diretamente alcançáveis por densidade a partir de x_1, ou seja, todos os pontos na $N_\varepsilon(x_1)$.

$$C_1 = \{x_1, x_5, x_6\}$$

O ponto x_1 é removido do conjunto C_a que agora irá ficar da seguinte maneira:

$$C_a = \{x_2, x_3, x_4, x_5, x_6, x_7\}$$

A ε-vizinhança de cada ponto não visitado no *cluster* C_1 é recuperada pelo algoritmo, com a finalidade de tentar inserir novos pontos ao *cluster*. Verificando inicialmente a $N_\varepsilon(x_5)$:

$$dist\ (x_5, x_1) = 0{,}2$$
$$dist\ (x_5, x_2) = 0{,}7$$
$$dist\ (x_5, x_3) = 0{,}7$$
$$dist\ (x_5, x_4) = 0{,}9$$
$$dist\ (x_5, x_6) = 0{,}4$$
$$dist\ (x_5, x_7) = 3{,}6$$

Verificou-se que o ponto x_5 não possui o número mínimo de pontos η em sua ε-vizinhança e, portanto nenhum ponto é diretamente alcançável por densidade a partir do ponto x_5. Nenhum novo ponto é adicionado ao *cluster* e o ponto x_5 é removido do conjunto de pontos não visitados, que fica da seguinte maneira.

$$C_a = \{x_2, x_3, x_4, x_6, x_7\}$$

O próximo ponto em C_1 a ter sua ε-vizinhança verificada é x_6:

$$dist\,(x_6, x_1) = 0{,}2$$
$$dist\,(x_6, x_2) = 0{,}9$$
$$dist\,(x_6, x_3) = 0{,}9$$
$$dist\,(x_6, x_4) = 1{,}1$$
$$dist\,(x_6, x_5) = 0{,}4$$
$$dist\,(x_6, x_7) = 3{,}2$$

A $N_\varepsilon(x_6)$ não possui a quantidade suficiente de pontos para tornar x_6 um ponto central no *cluster* e, portanto nenhum novo ponto é adicionado a C_1 que é finalizado:

$$C_1 = \{x_1, x_5, x_6\}$$

O conjunto auxiliar é atualizado:

$$C_a = \{x_2, x_3, x_4, x_7\}$$

O próximo passo do algoritmo consiste em visitar o próximo ponto não visitado da base de dados, ou seja, x_2. Recuperando a $N_\varepsilon(x_2)$:

$$dist\,(x_2, x_3) = 0{,}4$$
$$dist\,(x_2, x_4) = 0{,}4$$
$$dist\,(x_2, x_7) = 3{,}1$$

O ponto x_2 é marcado como *outlier*, pois a sua ε-vizinhança não possui o número mínimo de pontos η necessários para torná-lo um ponto central, e além disso, não está inserido na ε-vizinhança de nenhum outro ponto.

O conjunto C_a é atualizado:

$$C_a = \{x_3, x_4, x_7\}$$

A $N_\varepsilon(x_3)$ é recuperada:

$$dist\,(x_3, x_2) = 0{,}4$$
$$dist\,(x_3, x_4) = 0{,}2$$
$$dist\,(x_3, x_7) = 3{,}5$$

Como a $N_\varepsilon(x_3)$ não possui o número mínimo de pontos η, então esse ponto também é marcado como *outlier* e o conjunto C_a é atualizado:

$$C_a = \{x_4, x_7\}$$

Verificando a $N_\varepsilon(x_4)$:

$$dist\,(x_4, x_2) = 0{,}4$$
$$dist\,(x_4, x_3) = 0{,}2$$
$$dist\,(x_4, x_7) = 3{,}5$$

A $N_\varepsilon(x_4)$ também não possui o número mínimo de pontos η e x_4 também é marcado como *outlier*. O conjunto C_a é atualizado:

$$C_a = \{x_7\}$$

Verificando a $N_\varepsilon(x_7)$:

$$dist\,(x_7, x_2) = 3{,}1$$
$$dist\,(x_7, x_3) = 3{,}5$$
$$dist\,(x_7, x_4) = 3{,}5$$

Após verificação da ε-vizinhança de x_7, esse ponto também é marcado como *outlier*, por não possuir o número mínimo de pontos em sua ε-vizinhança e por não estar contido em nenhum *cluster*. O algoritmo remove x_7 de C_a:

$$C_a = \{\text{ø}\}$$

O conjunto C_a fica vazio, o que significa que todos os pontos da base de dados foram verificados e o algoritmo DBSCAN encerra a sua execução. Um *cluster* foi encontrado e quatro pontos foram classificados como *outlier*:

$$C_1 = \{x_1, x_5, x_6\}$$

$$Outliers = \{x_2, x_3, x_4, x_7\}$$

APÊNDICE C – MODELAGEM MATEMÁTICA UTILIZANDO A DISTÂNCIA EUCLIDIANA NORMALIZADA

Neste apêndice é demonstrada à modelagem matemática do algoritmo DBSCAN utilizando a distância euclidiana normalizada. Para demonstração foi utilizada uma base de dados contendo 7 elementos (x_1, x_2, x_3, x_4, x_5, x_6, x_7) com 2 atributos cada um (a, b). A Tabela 24 exemplifica os valores utilizados:

Tabela 24. Base de dados usada na modelagem

Atributos	x_1	x_2	x_3	x_4	x_5	x_6	x_7
a	5,1	4,9	4,7	4,6	5,0	5,3	8,0
b	3,5	3,0	3,2	3,1	3,6	3,5	3,0

Os parâmetros de entrada foram definidos da seguinte maneira:

e) **raio da ε-vizinhança de um ponto:** especifica o valor para a ε-vizinhança de um objeto. Foi definido o valor 0,3;

f) **número mínimo de pontos (η):** parâmetro que específica o número mínimo de pontos, no dado raio de ε-vizinhança, que um ponto precisa possuir para ser considerado um ponto central. Definiu-se o valor 3.

Primeiramente a base de dados deve ser normalizada, usando para isso, a normalização MIN-MAX. Os atributos foram normalizados para o intervalo entre zero e um:

$$z_i = \frac{v_i - \min_{vi}}{\max_{vi} - \min_{vi}}(n\_\max - n\_\min) + n\_\min$$

$$x_{1a} = \frac{5{,}1 - 4{,}6}{8{,}0 - 4{,}6}(1-0) + 0 = 0{,}1471 \qquad x_{1b} = \frac{3{,}5 - 3{,}0}{3{,}6 - 3{,}0}(1-0) + 0 = 0{,}8333$$

$$x_{2a} = \frac{4{,}9 - 4{,}6}{8{,}0 - 4{,}6}(1-0) + 0 = 0{,}0882 \qquad x_{2b} = \frac{3{,}0 - 3{,}0}{3{,}6 - 3{,}0}(1-0) + 0 = 0{,}0000$$

$$x_{3a} = \frac{4{,}7 - 4{,}6}{8.0 - 4.6}(1-0) + 0 = 0{,}0294 \qquad x_{3b} = \frac{3{,}2 - 3{,}0}{3{,}6 - 3{,}0}(1-0) + 0 = 0{,}3333$$

$$x_{4a} = \frac{4,6 - 4,6}{8.0 - 4,6}(1-0) + 0 = 0,0000 \qquad x_{4b} = \frac{3,1 - 3,0}{3,6 - 3,0}(1-0) + 0 = 0,1667$$

$$x_{5a} = \frac{5,0 - 4,6}{8,0 - 4,6}(1-0) + 0 = 0,1176 \qquad x_{5b} = \frac{3,6 - 3,0}{3,6 - 3,0}(1-0) + 0 = 1,0000$$

$$x_{6a} = \frac{5,3 - 4,6}{8,0 - 4,6}(1-0) + 0 = 0,2059 \qquad x_{6b} = \frac{3,5 - 3,0}{3,6 - 3,0}(1-0) + 0 = 0,8333$$

$$x_{7a} = \frac{8,0 - 4,6}{8,0 - 4,6}(1-0) + 0 = 1,0000 \qquad x_{7b} = \frac{3,0 - 3,0}{3,6 - 3,0}(1-0) + 0 = 0,0000$$

A base de dados irá ficar da seguinte maneira:

Tabela 25. Base de dados normalizada

Atributos	x_1	x_2	x_3	x_4	x_5	x_6	x_7
a	0,1471	0,0882	0,0294	0,0000	0,1176	0,2059	1,0000
b	0,8333	0,0000	0,3333	0,1667	1,0000	0,8333	0,0000

Com os valores normalizados, a distância euclidiana é aplicada e a matriz de dissimilaridade é montada:

$$d(x_{i,}x_j) = \sqrt{\sum_{l=1}^{d}(x_{il} - x_{jl})^2}$$

$$dist(x_1, x_2) = \sqrt{(0,1471 - 0,0882)^2 + (0,8333 - 0,0000)^2} = \sqrt{0,0035 + 0,6944} = 0,8354$$

$$dist(x_1, x_3) = \sqrt{(0,1471 - 0,0294)^2 + (0,8333 - 0,3333)^2} = \sqrt{0,0139 + 0,2500} = 0,5137$$

$$dist(x_1, x_4) = \sqrt{(0,1471 - 0,0000)^2 + (0,8333 - 0,1667)^2} = \sqrt{0,0217 + 0,4444} = 0,6826$$

$$dist(x_1, x_5) = \sqrt{(0,1471 - 0,1176)^2 + (0,8333 - 1,0000)^2} = \sqrt{0,0009 + 0,0278} = 0,1694$$

$$dist(x_1, x_6) = \sqrt{(0,1471 - 0,2059)^2 + (0,8333 - 0,8333)^2} = \sqrt{0,0035 + 0,0000} = 0,0592$$

$$dist(x_1, x_7) = \sqrt{(0,1471 - 1,0000)^2 + (0,8333 - 0,0000)^2} = \sqrt{0,7274 + 0,6944} = 1,1924$$

$$dist(x_2, x_3) = \sqrt{(0,0882 - 0,0294)^2 + (0,0000 - 0,3333)^2} = \sqrt{0,0035 + 0,1111} = 0,3385$$

$$dist(x_2, x_4) = \sqrt{(0,0882 - 0,0000)^2 + (0,0000 - 0,1667)^2} = \sqrt{0,0078 + 0,0278} = 0,1887$$

$$dist(x_2, x_5) = \sqrt{(0,0882 - 0,1176)^2 + (0,0000 - 1,0000)^2} = \sqrt{0,0009 + 1,0000} = 1,0004$$

$$dist(x_2, x_6) = \sqrt{(0,0882 - 0,2059)^2 + (0,0000 - 0,8333)^2} = \sqrt{0,0139 + 0,6944} = 0,8416$$

$$dist(x_2, x_7) = \sqrt{(0{,}0882 - 1{,}0000)^2 + (0{,}0000 - 0{,}0000)^2} = \sqrt{0{,}8314 + 0{,}0000} = 0{,}9118$$

$$dist(x_3, x_4) = \sqrt{(0{,}0294 - 0{,}0000)^2 + (0{,}3333 - 0{,}1667)^2} = \sqrt{0{,}0009 + 0{,}0278} = 0{,}1694$$

$$dist(x_3, x_5) = \sqrt{(0{,}0294 - 0{,}1176)^2 + (0{,}3333 - 1{,}0000)^2} = \sqrt{0{,}0078 + 0{,}4445} = 0{,}6725$$

$$dist(x_3, x_6) = \sqrt{(0{,}0294 - 0{,}2059)^2 + (0{,}3333 - 0{,}8333)^2} = \sqrt{0{,}0312 + 0{,}2500} = 0{,}5303$$

$$dist(x_3, x_7) = \sqrt{(0{,}0294 - 1{,}0000)^2 + (0{,}3333 - 0{,}0000)^2} = \sqrt{0{,}9421 + 0{,}1111} = 1{,}0263$$

$$dist(x_4, x_5) = \sqrt{(0{,}0000 - 0{,}1176)^2 + (0{,}1667 - 1{,}0000)^2} = \sqrt{0{,}0138 + 0{,}6944} = 0{,}8415$$

$$dist(x_4, x_6) = \sqrt{(0{,}0000 - 0{,}2059)^2 + (0{,}1667 - 0{,}8333)^2} = \sqrt{0{,}0424 + 0{,}4444} = 0{,}6977$$

$$dist(x_4, x_7) = \sqrt{(0{,}0000 - 1{,}0000)^2 + (0{,}1667 - 0{,}0000)^2} = \sqrt{1{,}0000 + 0{,}0278} = 1{,}0138$$

$$dist(x_5, x_6) = \sqrt{(0{,}1176 - 0{,}2059)^2 + (1{,}0000 - 0{,}8333)^2} = \sqrt{0{,}0078 + 0{,}0278} = 0{,}1887$$

$$dist(x_5, x_7) = \sqrt{(0{,}1176 - 1{,}0000)^2 + (1{,}0000 - 0{,}0000)^2} = \sqrt{0{,}7786 + 1{,}0000} = 1{,}3336$$

$$dist(x_6, x_7) = \sqrt{(0{,}2059 - 1{,}0000)^2 + (0{,}8333 - 0{,}0000)^2} = \sqrt{0{,}6306 + 0{,}6944} = 1{,}1511$$

Calculadas as distâncias, a matriz de dissimilaridade fica da seguinte maneira:

$$\begin{bmatrix} 0 & 0{,}8354 & 0{,}5137 & 0{,}6826 & 0{,}1694 & 0{,}0592 & 1{,}1924 \\ 0{,}8354 & 0 & 0{,}3385 & 0{,}1887 & 1{,}0004 & 0{,}8416 & 0{,}9118 \\ 0{,}5137 & 0{,}3385 & 0 & 0{,}1694 & 0{,}6725 & 0{,}5303 & 1{,}0263 \\ 0{,}6826 & 0{,}1887 & 0{,}1694 & 0 & 0{,}8415 & 0{,}6977 & 1{,}0138 \\ 0{,}1694 & 1{,}0004 & 0{,}6725 & 0{,}8415 & 0 & 0{,}1887 & 1{,}3336 \\ 0{,}0592 & 0{,}8416 & 0{,}5303 & 0{,}6977 & 0{,}1887 & 0 & 1{,}1511 \\ 1{,}1924 & 0{,}9118 & 1{,}0263 & 1{,}0138 & 1{,}3336 & 1{,}1511 & 0 \end{bmatrix}$$

Todos os pontos da base de dados são colocados no conjunto auxiliar:

$$C_a = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$$

O algoritmo escolhe o primeiro ponto no conjunto C_a, ou seja, x_1. Então é verifica a $N_\varepsilon(x_1)$ para saber se a cardinalidade do ponto é igual ou excede η, ou seja, verificar a condição: $Card\ (N_\varepsilon(x_1)) \geq \eta$. Portanto, o algoritmo DBSCAN verifica a matriz de dissimilaridade a fim de saber quantos pontos a ε-vizinhança de x_1 contêm:

$$dist(x_1, x_2) = 0{,}8354$$
$$dist(x_1, x_3) = 0{,}5137$$
$$dist(x_1, x_4) = 0{,}6826$$

$$dist\,(x_1, x_5) = 0{,}1694$$
$$dist\,(x_1, x_6) = 0{,}0592$$
$$dist\,(x_1, x_7) = 1{,}1924$$

Constata-se que os pontos x_5 e x_6 estão na ε-vizinhança de x_1 visto que $dist(x_1, x_5) \leq \varepsilon$ e $dist(x_1, x_6) \leq \varepsilon$ para ε= 0,3 e, portanto são diretamente alcançáveis por densidade a partir de x_1. Isso implica que x_1 é um ponto central, pois a $N_\varepsilon(x_1) \geq \eta$ .Então um *cluster* é formado contendo esse ponto e todos os pontos diretamente alcançáveis por densidade a partir de x_1, ou seja, todos os pontos na $N_\varepsilon(x_1)$.

$$C_1 = \{x_1, x_5, x_6\}$$

O ponto x_1 e removido do conjunto C_a:

$$C_a = \{x_2, x_3, x_4, x_5, x_6, x_7\}$$

Cada ponto no *cluster* C_1 tem sua ε-vizinhança verificada, com o objetivo de tentar inserir novos pontos ao *cluster* criado. Verificando inicialmente a $N_\varepsilon(x_5)$:

$$dist\,(x_5, x_1) = 0{,}1694$$
$$dist\,(x_5, x_2) = 1{,}0004$$
$$dist\,(x_5, x_3) = 0{,}6725$$
$$dist\,(x_5, x_4) = 0{,}8415$$
$$dist\,(x_5, x_6) = 0{,}1887$$
$$dist\,(x_5, x_7) = 1{,}3336$$

A $N_\varepsilon(x_5)$ possui o número mínimo de pontos η em sua ε-vizinhança e, portanto x_5 é também um ponto central no *cluster* C_1. Como todos os pontos diretamente alcançáveis por densidade a partir de x_5 já estão inseridos no *cluster*, nenhum novo ponto é adicionado e o ponto x_5 é removido do conjunto de pontos não visitados.

$$C_a = \{x_2, x_3, x_4, x_6, x_7\}$$

O próximo ponto em C_1 a ter sua ε-vizinhança verificada é x_6:

$$dist\,(x_6, x_1) = 0{,}0592$$
$$dist\,(x_6, x_2) = 0{,}8416$$

$$dist\,(x_6, x_3) = 0{,}5303$$
$$dist\,(x_6, x_4) = 0{,}6977$$
$$dist\,(x_6, x_5) = 0{,}1887$$
$$dist\,(x_6, x_7) = 1{,}1511$$

Os pontos x_1 e x_5 são diretamente alcançáveis por densidade a partir de x_6 e, portanto o ponto x_6 também é um ponto central em C_1. Porém, os pontos na ε-vizinhança de x_1 já pertencem ao *cluster* e novamente nenhum novo ponto é adicionado. Assim, um primeiro *cluster* e encontrado é finalizado.

$$C_1 = \{x_1, x_5, x_6\}$$

O conjunto auxiliar é atualizado e o algoritmo DBSCAN continua a pesquisa:

$$C_a = \{x_2, x_3, x_4, x_7\}$$

Verificando o conjunto C_a, o próximo ponto a ter a sua ε-vizinhança recuperada é x_2. Recuperando a $N_\varepsilon(x_2)$

$$dist\,(x_2, x_3) = 0{,}3385$$
$$dist\,(x_2, x_4) = 0{,}1887$$
$$dist\,(x_2, x_7) = 0{,}9118$$

O ponto x_2 é marcado como *outlier*, pois a sua ε-vizinhança não possui o número mínimo de pontos η necessários para torná-lo um ponto central.

O conjunto C_a é atualizado:

$$C_a = \{x_3, x_4, x_7\}$$

A $N_\varepsilon(x_3)$ é recuperada:

$$dist\,(x_3, x_2) = 0{,}3385$$
$$dist\,(x_3, x_4) = 0{,}1694$$
$$dist\,(x_3, x_7) = 1{,}0263$$

A $N_\varepsilon(x_3)$ não possui o número mínimo de pontos η, então esse ponto é marcado como *outlier* e o conjunto C_a é atualizado:

$$C_a = \{x_4, x_7\}$$

Verificando a $N_\varepsilon(x_4)$:

$$dist(x_4, x_2) = 0{,}1887$$
$$dist(x_4, x_3) = 0{,}1694$$
$$dist(x_4, x_7) = 1{,}0138$$

A $N_\varepsilon(x_4)$ contêm o número mínimo de pontos η, portanto esse ponto e toda a sua ε-vizinhança são colocados em um *cluster*.

$$C_2 = \{x_2, x_3, x_4\}$$

O conjunto C_a é atualizado:

$$C_a = \{x_7\}$$

Como não existem mais pontos suficientes em C_a para formar um *cluster* e, não é necessário verificar a ε-vizinhança de x_7. O algoritmo classifica esse ponto como *outlier* e remove x_7 de C_a:

$$C_a = \{\emptyset\}$$

O conjunto C_a fica vazio, o que significa que todos os pontos da base de dados foram visitados e o algoritmo DBSCAN encerra a sua execução. Dois *clusters* foram encontrados e um ponto foi classificado como *outlier*:

$$C_1 = \{x_1, x_5, x_6\}$$
$$C_2 = \{x_2, x_3, x_4\}$$
$$Outlier = \{x_7\}$$

APÊNDICE D – ANÁLISE DOS DADOS

As medidas de tendência central são valores que representam a distribuição, como por exemplo, a média aritmética, mediana e moda (BISQUERRA; SARRIERA; MARTINEZ, 2004).

A média aritmética representa o valor médio de uma distribuição. Sejam *n* observações efetivas de uma variável aleatória, então a média aritmética pode ser definida como (BARBETTA; REIS; BORNIA, 2010):

$$\bar{x} = \frac{x_1 + x_2 + .. + x_n}{n} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

A mediana é o valor que ocupa o lugar central de uma série de valores ordenados. Se o número de elementos for ímpar, a mediana coincide com o elemento que ocupa a posição mais central. Se o número de elementos for par, a mediana será a média aritmética dos elementos que ocupam o lugar mais central na distribuição (BISQUERRA; SARRIERA; MARTINEZ, 2004).

A moda é o valor que mais se repete em uma série de valores. Uma distribuição pode ser unimodal, quando tem somente uma moda, bimodal quando tem duas modas, multimodal se tem mais de duas e amodal se não possui nenhuma moda (BISQUERRA; SARRIERA; MARTINEZ, 2004).

A forma mais simples de se medir a dispersão dos valores no conjunto de dados observado é por meio da amplitude. A amplitude é resultante da diferença entre o maior e menor valor no conjunto de dados. Matematicamente pode ser definida como (BARBETTA; REIS; BORNIA, 2010):

$$a = \max(x_1, x_2, ... x_n) - \min(x_1, x_2, ... x_n)$$

A variância e o desvio padrão são medidas mais apropriadas para mensuração da dispersão, pois a amplitude considera somente os dois valores mais extremos. Tanto a variância quanto o desvio padrão levam em consideração o desvio de cada valor com relação à média aritmética. A variância pode ser definida como a média aritmética dos desvios quadráticos (BARBETTA; REIS; BORNIA, 2010):

$$s^2 = \frac{1}{n-1} = \sum_{i=1}^{n} \left(x_i - \bar{x} \right)^2$$

O desvio padrão corresponde à raiz quadrada positiva da variância, uma vez que a variância é calculada em função dos desvios quadráticos. Desse modo, o desvio padrão pode ser expresso na mesma unidade de medida dos dados em que se pretende realizar a análise (BARBETTA; REIS; BORNIA, 2010):

$$s = \sqrt{\frac{1}{n-1} = \sum_{i=1}^{n} \left(x_i - \bar{x} \right)^2}$$

Quando se deseja conhecer outros aspectos relativos ao conjunto de dados, além de tendências de medida central e de variância, podem ser utilizados os quartis. Os quartis dividem um conjunto de dados em quatro partes que contêm cada uma 25% dos dados da série (BARBETTA; REIS; BORNIA, 2010; GALVANI; LUCHIARI, 2004).

O quartil inferior (q_i) é o valor que delimita os 25% menores valores. O quartil superior (q_s) é o valor que delimita os 25% maiores valores. O quartil mediano (m_d) separa os 50% menores valores dos 50% maiores valores. O desvio interquartílico (d_q) é dado por ($d_q = q_s - q_i$), sendo que em distribuições mais dispersas, os valores dos quartis ficam mais distantes. Na Figura 39 verifica-se a distribuição dos quartis em um conjunto de dados.

Em distribuições com simetria, a distância entre o quartil inferior e a mediana é igual à distância entre o quartil superior e a mediana. Em distribuições assimétricas, as distâncias entre esses quartis são diferentes (BARBETTA; REIS; BORNIA, 2010; FIELD,

2009; GALVANI; LUCHIARI, 2004).

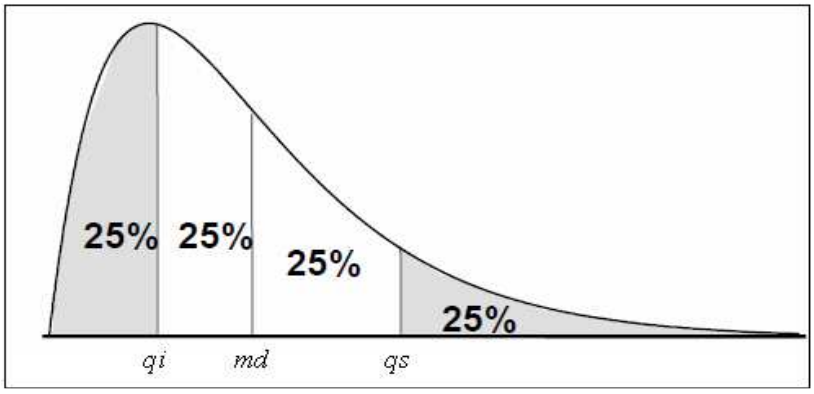

Figura 39. Quartis
Fonte:Adaptado de BARBETTA, P.; REIS, M.; BORNIA, A. (2010)

Utilizando os quartis a mediana e os extremos, é possível ter informações sobre a posição central, dispersão e assimetria de uma distribuição, sendo que uma forma de se apresentar graficamente esses conceitos é através do gráfico *boxplot* (BARBETTA; REIS; BORNIA, 2010).

Esse gráfico é montado utilizando a mediana da série de valores, o valor de maior magnitude da série, o valor de menor magnitude da série e os quartis. Com o *boxplot* é possível representar o desvio interquartílico. O *boxplot* é um retângulo que representa a faixa dos 50% valores mais típicos da distribuição, sendo dividido no valor que corresponde a mediana da série, indicando assim, o quartil inferior, o superior e a mediana (BARBETTA; REIS; BORNIA, 2010; FIELD, 2009).

Entre os quartis e os valores extremos são traçadas linhas. Essas linhas inferiores e superiores se estendem, respectivamente, do quartil inferior (q_i) até o menor valor não inferior a q_i - 1.5I d_q e do quartil superior (q_s) até o maior valor não superior a q_s + 1.5dq. Os valores inferiores a q_i - 1.5I d_q e superiores a q_s + 1.5I d_q são representados individualmente no gráfico e são caracterizados como *outliers* (Figura 40) (BARBETTA; REIS; BORNIA, 2010; FIELD, 2009).

Outliers são valores que diferem excessivamente do seu conjunto e costumam

distorcer os resultados das análises estatísticas. Uma forma de identificar a presença de *outliers* no conjunto de dados é através do gráfico de *boxplot*, que permite representar a distribuição de um conjunto de dados com base em alguns de seus parâmetros descritivos (BARBETTA; REIS; BORNIA, 2010; FIELD, 2009; GALVANI; LUCHIARI, 2004).

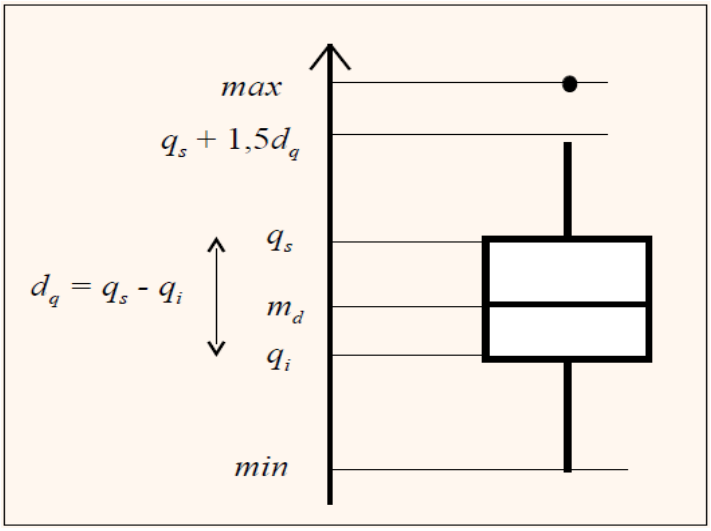

Figura 40. Construção de um gráfico do tipo *boxplot*
Fonte: BARBETTA, P.; REIS, M.; BORNIA, A. (2010)

Em muitas provas paramétricas de comparação entre médias, é exigido que os conjuntos de dados envolvidos provenham de uma distribuição normal[39]. Por esse motivo, antes da aplicação desse tipo de prova, costumeiramente se realizam testes para verificar a hipótese de normalidade na distribuição de valores. O teste de *Kolmogorov-Smirnov* é o mais utilizado para esse fim (BISQUERRA; SARRIERA; MARTINEZ, 2004).

O teste de *Kolmogorov-Smirnov* tem como objetivo verificar se os dados de uma amostra comportam-se de acordo com uma distribuição teórica, ou seja, verificar se a amostra observada *X* provém de uma distribuição teórica *D*. Quando as diferenças entre essas duas distribuições superam valores pré-estabelecidos, a hipótese de aderência dos dados a distribuição teórica é descartada (BISQUERRA; SARRIERA; MARTINEZ, 2004; FIELD, 2009). As hipóteses para esse teste são (BARBETTA; REIS; BORNIA, 2010):

[39] A distribuição normal é caracterizada por uma função de probabilidade, cujo gráfico descreve uma curva em forma de sino. Essa forma de distribuição evidência que existe maior probabilidade de a variável aleatória assumir valores próximos do centro (BARBETTA; REIS; BORNIA, 2010).

H_0: os dados da amostra *X* tem uma determinada distribuição *D*;

H_1: *X* não possui a mesma distribuição que *D* (não há aderência).

Onde:

a) $H_{0:}$ hipótese nula;

b) $H_{1:}$ hipótese alternativa;

c) *X*: amostra da população observada;

d) *D*: distribuição teórica, que pode ser uma distribuição clássica de probabilidades (normal, exponencial, binomial, uniforme).

Esse teste utiliza a distribuição de freqüência acumulada que ocorreria dada a distribuição teórica, e a compara com a distribuição de freqüências acumulada observada, sendo que a distribuição teórica representa o que seria esperado sob a hipótese nula (H_0) e então é verificado se as distribuições teórica e observada mostram divergência (BARBETTA; REIS; BORNIA, 2010; BISQUERRA; SARRIERA; MARTINEZ, 2004).

Se o valor absoluto da maior das diferenças obtidas é não significativo, ou seja, o *p-value*[40] é superior a um nível de significância[41] pré-estabelecido, então é sugerido que os dados observados não diferem significativamente da distribuição teórica e o teste aceita a hipótese nula. Pelo contrario, caso o *p-value* for significativo, com valor inferior ao pré-estabelecido então existem fortes evidências de que a amostra não é proveniente da distribuição teórica, levando o teste a rejeitar a hipótese nula e aceitar a hipótese alternativa (BARBETTA; REIS; BORNIA, 2010).

[40] O *p-value* pode ser definido como a probabilidade de a estatística do teste acusar um resultado tão ou mais distante do esperado (BARBETTA; REIS; BORNIA, 2010).

[41] Quando se deseja confirmar ou rejeitar uma hipótese geralmente é estabelecida a probabilidade tolerável de ocorrência de um erro ao rejeitar a hipótese nula(H_0). Esse valor é conhecido como nível de significância do teste, designado pela letra grega α(alfa) (BARBETTA; REIS; BORNIA, 2010).

Quando os dados não seguem a tendência da distribuição normal, são utilizados testes não-paramétricos[42]. Esses testes são utilizados para comprovar se as diferenças entre as médias observados em dois grupos independentes são estatisticamente significativas (BISQUERRA; SARRIERA; MARTINEZ, 2004).

Para comparar duas médias de populações provenientes de amostras independentes um dos testes não-paramétricos mais utilizados é o de *Mann-Whitney*, que destina-se a verificar se duas amostras independentes provêm de populações com médias iguais, em nível de significância pré-estabelecido.

A posição central das populações é comparada com base em amostras independentes, extraídas aleatoriamente dessas populações. As hipóteses do teste podem ser colocadas como (BARBETTA; REIS; BORNIA, 2010; BISQUERRA; SARRIERA; MARTINEZ, 2004):

$$H_0 : \eta_1 = \mu_2 \text{ e } H_1 : \eta_1 \neq \eta_2$$

Onde:

a) $H_{0:}$ hipótese nula. Não existem diferenças entre as medianas das duas populações;

b) $H_{1:}$ hipótese alternativa. Existem diferenças significativas entre as medianas das duas populações;

c) $\eta_{1:}$ mediana da primeira população;

d) $\eta_{2:}$ mediana da segunda população.

Sejam n_1 e n_2 os tamanhos das amostras 1 e 2. Os $n_1 + n_2$ elementos devem ser combinados, identificados e ordenados de forma crescente com relação à variável observada, e são atribuídos o posto 1 ao menor valor , posto 2 ao segundo menor valor e assim por

[42] Testes não-paramétricos são aplicados sem a necessidade de se fazer qualquer tipo de suposição sobre as distribuições e a origem das variáveis que estão sendo estudadas (BISQUERRA; SARRIERA; MARTINEZ, 2004).

diante, até o posto $n_1 + n_2$.

O teste é desenvolvido com base na soma dos postos ocupados na amostra ordenada. Se a hipótese nula (H_0) for verdadeira, a média dos postos em cada um dos dois grupos será quase a mesma, e se a soma dos postos para um grupo é muito grande ou muito pequena, pode-se suspeitar que as amostras não foram extraídas da mesma população.

A estatística do teste é definida por:

$$U = w_1 - \left[\frac{n_1(n_1+1)}{2}\right]$$

Levando em consideração o nível de significância adotado, o valor encontrado pela estatística do teste U deve ser comparado com um valor crítico tabelado pelo próprio teste estatístico, e caso a hipótese nula (H_0) for verdadeira, a estatística do teste irá apresentar um valor menor que o valor crítico tabelado (BARBETTA; REIS; BORNIA, 2010; BISQUERRA; SARRIERA; MARTINEZ, 2004).

APÊNDICE E – ARTIGO

O Método de Densidade pelo Algoritmo *Density-Based Spatial Clustering of Applications with Noise* (DBSCAN) na Tarefa de Clusterização na *Shell* Orion *Data Mining*

Éverton Marangoni Gava[1], Merisandra Côrtes de Mattos[2]

evertonmgava@yahoo.com.br, mem@unesc.net

[1] Acadêmico do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma– SC

[2] Professora do Curso de Ciência da Computação – Unidade Acadêmica de Ciências, Engenharias e Tecnologias – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma– SC

Resumo. *Progressos constantes no processo de aquisição e armazenamento digital de informações permitiram que as mais variadas instituições formassem grandes repositórios de dados. Portanto, torna-se necessário o desenvolvimento e o aperfeiçoamento de tecnologias destinadas à análise de informações, possibilitando a obtenção de novos conhecimentos. Dentre essas tecnologias, destaca-se o data mining, que por meio da aplicação de algoritmos com finalidades específicas, tenta extrair uma série de padrões possivelmente existentes em um conjunto de dados. Este artigo demonstra a modelagem matemática e o desenvolvimento do algoritmo DBSCAN, que utiliza o método de densidade para a tarefa de clusterização. Esta tarefa é responsável por agrupar uma base de dados considerando a semelhança entre os dados, sendo que o método de densidade torna possível a obtenção de clusters com formatos arbitrários e consegue ser robusto na presença de outliers.*

1. Introdução

O progresso nas tecnologias de armazenamento e aquisição de dados resultou em crescimento das bases de dados. Embora sabendo que padrões interessantes e informações potencialmente úteis podem ser extraídas desses repositórios, o volume de dados torna difícil, senão impraticável, a busca por esse conhecimento implícito sem o auxílio de tecnologias computacionais [HAN; KAMBER, 2006].

A fim de facilitar o processo de descoberta de conhecimento, técnicas capazes de extrair informações significativas destes vastos repositórios foram desenvolvidas, sendo que a procura por relações úteis entre os dados ficou conhecida como *Knowledge Discovery in Databases* (KDD), tendo o *data mining* como a principal etapa desse processo.

Os algoritmos de *data mining* geralmente são disponibilizados em ferramentas computacionais específicas para esse fim, sendo que existe em desenvolvimento a *Shell* Orion *Data Mining Engine*, um projeto acadêmico implementado pelo Grupo de Pesquisa em Inteligência Computacional Aplicada, do Curso de Ciência da Computação, na Universidade do Extremo Sul Catarinense. Considerando as diversas tarefas de *data mining* existentes, a *Shell Orion* atualmente contempla as tarefas de classificação,

associação e clusterização.

A clusterização pode ser vista como uma das tarefas básicas no processo de *data mining*, permitindo identificar os grupos existentes em um conjunto de dados, assim auxiliando na estruturação e na compreensão do mesmo. O método de densidade para a clusterização destaca-se, pois permite que sejam encontrados *clusters* com formatos arbitrários e é relativamente robusto na presença de *outliers* (dados que destoam do padrão geral da base) entre os objetos de uma base de dados.

Desta forma, neste artigo apresenta-se o desenvolvimento do método baseado em densidade por meio do algoritmo DBSCAN, no módulo de clusterização da *Shell Orion*.

2. *Data Mining*

O *data mining* pode ser definido como o processo de explorar e analisar grandes conjuntos de dados, extraindo informação e conhecimento sob a forma de novas relações e padrões úteis na resolução de problemas de um domínio de aplicação específico.

Considerando que o *data mining* pode ser aplicado em diversos campos de pesquisa, e que os usuários do conhecimento gerado por esse processo podem estar interessados em tipos distintos de padrões, naturalmente existem diversas tarefas de *data mining*, sendo que a escolha de uma determinada tarefa depende do conhecimento que se deseja obter [WITTEN; FRANK; HALL, 2011].
De modo geral, as principais tarefas de *data mining* são:

a) **classificação**: categoriza registros em classes pré-definidas;

b) **associação:** procura descrever as relações de associação entre diferentes itens de uma transação na base de dados;

c) **clusterização**: consiste na separação de um conjunto de dados em grupos de objetos similares.

3. A Tarefa de Clusterização no Processo de *Data Mining*

A tarefa de segmentação de um grupo heterogêneo de dados em vários subgrupos, também chamados de *clusters*, é denominada clusterização. Diferentemente da tarefa de classificação, na clusterização não existem classes pré-definidas ou exemplos, onde os registros são agrupados conforme a sua similaridade com os demais dados, sendo que o principal objetivo dessa tarefa é encontrar uma estrutura conveniente e válida de um conjunto de dados [HAN; KAMBER, 2006].

Um *cluster* pode ser entendido como uma coleção de registros que são similares entre si e dissimilares de objetos em outros grupos, onde objetos pertencentes a um dado *cluster* devem compartilhar um conjunto de propriedades comuns, sendo que essas propriedades não são compartilhadas com objetos de outros *clusters* [FAYADD; PIATETSKY-SHAPIRO, SMYTH, 1996].

Além de permitir a estruturação e fornecer uma melhor compreensão do conjunto de dados original, os resultados da tarefa de clusterização também podem ser utilizados por outras técnicas de *data mining*, que realizariam seu trabalho nos *clusters* encontrados.

3.1. O Método de Densidade na Tarefa de Clusterização

Conjuntos de dados reais usualmente apresentam agrupamentos com densidades e formas distintas, além de possuírem quantidade significativa de elementos considerados *outliers*,

tornando-se desejável a utilização de métodos eficientes para lidar com esse tipo de cenário. Considerando que grande parte dos métodos de clusterização geralmente encontra dificuldades para gerar *clusters* com formatos arbitrários, e além disso, não possuem a capacidade de encontrar *outliers* entre os registros de uma base de dados, foi proposto o método de clusterização baseado em densidade [KRIEGEL et al, 2011].

Algoritmos que utilizam essa abordagem de clusterização encontram agrupamentos baseados na densidade de elementos em uma determinada região no espaço de objetos. De acordo com o método baseado em densidade, um *cluster* pode ser definido como uma região densa no espaço de dados, que é separada de outras áreas densas, por regiões de baixa densidade, que representam *outliers*. Essas áreas podem ter um formato arbitrário e ainda os pontos dentro de uma região podem estar distribuídos arbitrariamente [HAN; KAMBER, 2006].

Existem diversos algoritmos que utilizam a abordagem baseada em densidade para realizar a tarefa de clusterização, sendo que cada um possui seus conceitos particulares para definir áreas com alta densidade de objetos no espaço de dados. Dentre esses algoritmos, o DBSCAN é considerado referência, pois foi o primeiro a utilizar o método de densidade para clusterização de dados [ESTER et al, 1996; KRIEGEL et al, 2011].

4. O Algoritmo DBSCAN

No ano de 1996, Martin Ester, Hans-Peter Kriegel, Jörg Sander e Xiaowei Xu publicaram o artigo intitulado *A Density Based Spatial Clustering of Applications With Noise*. Nesse artigo foi apresentado o algoritmo DBSCAN, que pode ser aplicado a grandes conjuntos de dados contaminados por *outliers*, ao mesmo tempo em que encontra *clusters* com diversos formatos e com eficiência aceitável.

O algoritmo DBSCAN encontra agrupamentos baseado na vizinhança dos objetos, onde a densidade associada a um ponto é obtida por meio da contagem do número de pontos vizinhos em uma determinada região ao redor desse ponto [ESTER et al, 1996].

Esse algoritmo possui a capacidade de encontrar *clusters* considerando as propriedades dos dados, pois não requer que seja informado antecipadamente o número de *clusters*, permitindo a formação de grupos com formatos arbitrários. Outras características importantes do algoritmo são a capacidade de identificar *outliers* e a possibilidade de poder trabalhar com diversas medidas de distância [ESTER et al, 1996].

5. O Algoritmo DBSCAN na Tarefa de Clusterização da *Shell Orion Data Mining Engine*

A modelagem do algoritmo DBSCAN iniciou-se com a construção dos diagramas de caso de uso, atividades e seqüência utilizando os padrões UML. Posteriormente foi desenvolvida a demonstração matemática do funcionamento do algoritmo com a finalidade de facilitar o entendimento e a sua implementação.

O algoritmo necessita de dois parâmetros de entrada:

c) **raio de ε-vizinhança de um ponto**: determina o raio de vizinhança ε para cada ponto da base de dados. Dado o parâmetro ε, o algoritmo DBSCAN verifica a quantidade de pontos contidos no raio ε para cada ponto da base de dados, e se essa quantidade exceder certo número, um *cluster* é formado;

a) **número mínimo de pontos (η)**: parâmetro que especifica o número mínimo de pontos, no dado raio de ε-vizinhança, que um ponto precisa possuir para ser

considerado um ponto central e conseqüentemente, de acordo com as definições de *cluster* baseado em densidade, iniciar a formação de um *cluster*.

Definidos os parâmetros de entrada e a base de dados a ser clusterizada, o primeiro passo do DBSCAN consiste em montar uma estrutura denominada matriz de dissimilaridade. Em uma matriz de dissimilaridade é possível representar a distância entre pares de objetos. Essa matriz sempre será quadrada e de tamanho *n* x *n*, onde *n* representa a quantidade de objetos que serão clusterizados (Figura 1).

$$\begin{bmatrix} 0 & & & \\ dist(2,1) & 0 & & \\ dist(3,1) & dist(3,2) & 0 & \\ dist(4,1) & dist(4,2) & dist(4,3) & 0 \end{bmatrix}$$

Figura 1 - Matriz de dissimilaridade

Essa matriz de dissimilaridade é descoberta empregando-se uma medida que é responsável pelo cálculo de distância para todos os pares de objetos da base de dados. As medidas de distancia mais comumente utilizadas são a distância euclidiana (equação 1) e a *Manhattan* (equação 2):

$$d(x_{i,}x_j) = \sqrt{\sum_{l=1}^{d} (x_{il} - x_{jl})^2} \quad (1)$$

$$d(x_{i,}x_j) = \sum_{l=1}^{d} \left| (x_{il} - x_{jl}) \right| \quad (2)$$

Sabendo que atributos diferentes podem ser medidos em escalas distintas, caso for utilizada diretamente uma medida de distância como a euclidiana, por exemplo, atributos com escalas maiores irão se sobrepor a atributos medidos em escalas menores, tornando a clusterização tendenciosa [HAN; KAMBER, 2006]. Para normalização das escalas dos atributos, utilizou-se a normalização MIN-MAX:

$$z_i = \frac{v_i - \min_{vi}}{\max_{vi} - \min_{vi}}(n\_\max - n\_\min) + n\_\min$$

Tendo a matriz de dissimilaridade montada, o próximo passo executado pelo algoritmo consiste em verificar a ε-vizinhança de cada ponto da base de dados *D* com a finalidade de identificar possíveis pontos centrais para iniciar a formação dos agrupamentos. Um objeto x_j está na ε-vizinhança de um objeto x_i se:

$$x_j \in N_\varepsilon(x_i) = \{ x_i, x_j \in D \mid dist(x_i, x_j) \leq \varepsilon \}$$

A matriz de dissimilaridade contendo as distâncias entre os pares de objetos da base de dados é consultada e se a distância entre um ponto x_i e um ponto x_j for menor ou igual a o dado parâmetro ε, ou seja, $dist(x_i, x_j) \leq \varepsilon$, então o ponto x_j está na ε-vizinhança do ponto x_i.

Verificados quais os objetos que estão contidos na vizinhança de x_i, o algoritmo precisa verificar a cardinalidade (*Card*) desse ponto com relação aos objetos vizinhos, com a finalidade de definir a condição do ponto.

Caso a cardinalidade de x_i com relação ao raio de ε-vizinhança seja igual ou exceda o parâmetro η, então esse ponto é considerado um ponto central e os objetos contidos em sua ε-vizinhança são então diretamente alcançáveis por densidade. Para que um ponto x_j

seja diretamente alcançável por densidade a partir de um ponto x_i, o objeto x_i deve satisfazer as condições:

$$x_j \in N_\varepsilon(x_i)$$
$$Card\,(N_\varepsilon(x_i)) \geq \eta$$

Se as duas condições forem satisfeitas, o algoritmo irá criar um cluster C_i contendo todos os pontos diretamente alcançáveis por densidade a partir de um determinado objeto x_i:

$$C_i = \{N_\varepsilon(x_i)\}$$

A partir do momento em que o DBSCAN forma um *cluster* C_i, todo o objeto nesse grupo tem recursivamente a sua ε-vizinhança recuperada permitindo assim que novos pontos possam ser adicionados ao *cluster*. O processo de consulta de vizinhos próximos é repetido até que todos os pontos em C_i sejam verificados. Quando isso ocorrer, o algoritmo encerra o crescimento de C_i e visita o próximo ponto não visitado da base de dados. O processo é repetido até que todos os registros do conjunto de objetos tenham sido visitados e classificados.

O algoritmo distingue três tipos de objetos em um conjunto de dados:

a) **pontos centrais:** são pontos que estão no interior de uma região densa,onde existem pelo menos η pontos no raio ε desse objeto. A cardinalidade desses pontos em relação ao parâmetro de ε-vizinhança deve ser de no mínimo η pontos;

b) **pontos de borda:** estão na fronteira de uma região densa, ou seja, sã pontos que estão na ε-vizinhança de algum ponto central, porém não são pontos centrais, pois a cardinalidade desses pontos em relação ao raio ε não excede η;

c) ***outliers*:** esses pontos não são centrais e nem de borda e assim não são conectados por densidade a nenhum outro ponto, não pertencendo a nenhum *cluster*.

Caso um ponto for classificado como *outlier* pelo algoritmo, posteriormente ele pode estar na ε-vizinhança de outro ponto não visitado ainda pelo DBSCAN. Sendo assim, essa classificação pode ser removida caso o objeto seja diretamente alcançável por densidade a partir de um ponto central ainda não visitado.

5.1. Implementação

O algoritmo DBSCAN foi desenvolvido no módulo de clusterização da *Shell Orion Data Mining Engine*, por meio da linguagem de programação Java e do ambiente de programação integrado *NetBeans* 7.0.1.

Para a execução do algoritmo DBSCAN é necessário que sejam informados os parâmetros de ε-vizinhança e o número mínimo de pontos η nessa vizinhança. Também existe a possibilidade de escolha da medida de distância que será utilizada, sendo que estão disponíveis a distância euclidiana e a *Manhattan*.

Para a determinação dos parâmetros de entrada necessários ao algoritmo é proposta em uma heurística simples, porém efetiva para boa parte dos casos para determinar os parâmetros globais de entrada do algoritmo [ESTER et al, 1996].

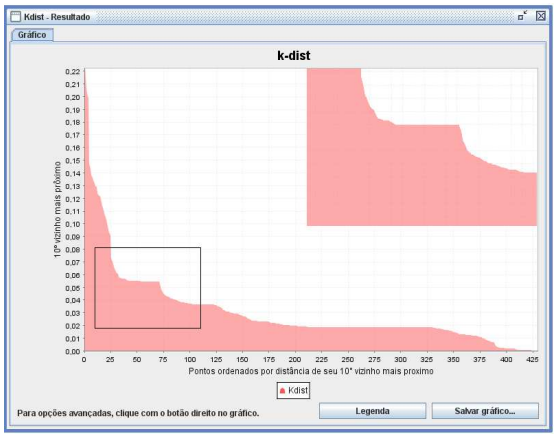

Figura 2. Heurística para determinação dos parâmetros de entrada

Visualizando o gráfico da função *k-dist* (Figura 2), o usuário poderá verificar no eixo *y* (ordenadas) que os valores apresentados dizem respeito às distâncias ordenadas decrescentemente entre um ponto e seu *k*-ésimo vizinho mais próximo. No eixo *x* (abscissas) estão os pontos da base de dados. É possível verificar que no eixo *y* entre os valores 0,04 e 0,06 ocorre repentinamente uma mudança brusca no gráfico. Portanto, todos os valores a direita desse "vale" estão contidos em algum *cluster* e todos os pontos mais a esquerda desse ponto podem ser considerados pontos de borda e *outliers*.

Assim, é possível definir para o parâmetro ε valores entre 0,04 e 0,06, e se define para o parâmetro η o valor *k* com o qual a função *k-dist* foi executada.

Os resultados podem ser visualizados em forma gráfica e em forma de resumo pela aba *Geral* (Figura 3). Também obtem-se os grupos encontrados em uma estrutura de dados na aba *Grupos* e exportar os resultados em um arquivo com formato *Strutured Query Language* (SQL), permitindo utilizar o resultado encontrado pelo processo como entrada de dados para outra tarefa de DM, como a classificação e a análise de *outliers*.

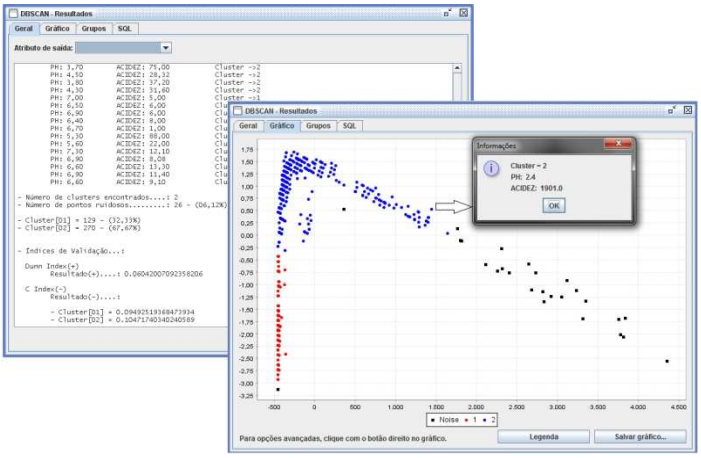

Figura 3. Resumo do resultado da clusterização

Como forma de auxílio na determinação da qualidade dos grupos encontrados por algoritmos de clusterização, o resultado do agrupamento deve ser validado com o objetivo de verificar a solução encontrada, sendo que essa tarefa é feita geralmente por meio de índices estatísticos. Foram implementados o índice de *Dunn* proposto por J.C. Dunn em 1974 e o *C-Index* proposto por L. J. Hubert e J. R. Levin em 1976.

O índice de *Dunn* permite à avaliação dos *clusters* com relação à compactação interna e separação externa, identificando o quanto os grupos encontrados são densos e

separados dos outros [BEZDEK, 2005]:

$$Dunn = \min_{1<i<k}\left(\min_{i+1\le j\le k}\left(\frac{dist\ (c_i, c_j)}{\max\limits_{1\le l\le k} diam\ (c_l)}\right)\right)$$

Já o *C-Index* permite que a homogeneidade de cada *cluster* encontrado seja avaliada separadamente, ou seja, com esse índice é possível verificar até que ponto, objetos similares foram colocados em um mesmo *cluster* [MILLIGAN; COOPER, 1985]:

$$C - Index = \frac{d_w(C_i) - \min\ (n_w)}{\max(\ n_w) - \min(\ n_w)}$$

Onde:

$$d_w(C_i) = \sum_{x_i, x_j \in C_i} dist\ (x_i, x_j)$$

5.2. Resultados Obtidos

A base de dados utilizada para avaliação do algoritmo DBSCAN apresenta indicadores ambientais da qualidade dos recursos hídricos da bacia do rio Araranguá, sendo composta por 679 registros com 20 atributos.

Para a seleção dos parâmetros de entrada do algoritmo o seguinte critério foi adotado:

f) os atributos de entrada desejados foram selecionados;

g) para o parâmetro η foram selecionados valores entre 4 e 10;

h) a função *k-dist* foi calculada iterativamente com *k* valendo η em cada iteração, ou seja *η-dist*;

i) o parâmetro ε foi definido iterativamente analisando-se o gráfico *k-dist*;

j) os mesmos passos foram repetidos usando as três medidas de distância implementadas no trabalho: euclidiana, euclidiana normalizada e *Manhattan*.

Os atributos de entrada selecionados foram os índices de pH e a concentração de ferro presentes nas amostras obtidas ao longo da bacia do rio Araranguá. O objetivo desse teste foi separar os registros coletados em dois grupos, ou seja, pontos de coleta em que a água está contaminada e pontos de coleta em que a água não está contaminada. Na Tabela 1 estão descritos os resultados gerados pelo algoritmo utilizando a distância euclidiana com a normalização dos dados.

Tabela 1- Resultados obtidos utilizando a distância euclidiana normalizada

Cluster	**Quantidade de elementos**	**Porcentagem**	**Situação do ponto de coleta**
1	132	19,44%	não-contaminado
2	538	79,23%	contaminado
Outliers	9	01,33%	-

Empregando-se a distância euclidiana normalizada o DBSCAN identificou que em 79,23% das amostras coletadas na bacia do rio Araranguá os índices de pH estão geralmente abaixo de 5.0 e a concentração de ferro presente nas águas se encontra em índices geralmente maiores que 20 mg/L. Baixos valores de pH e altas concentrações de ferro e outros sulfatos demonstram a degradação dos rios da bacia por atividades ligadas a extração

de carvão e fazem com que os seus recursos hídricos se apresentem de maneira imprópria para o consumo humano e uso em geral [ALEXANDRE; KREBS, 1996].

Utilizando-se a medida de distância *Manhattan* e euclidiana sem normalização o algoritmo não obteve resultados satisfatórios, classificando em um único *cluster* pontos que poderiam estar em grupos separados.

5.2.1. Avaliação dos Parâmetros de Entrada no Resultado do Algoritmo

Os parâmetros de entrada do DBSCAN foram avaliados com o objetivo de verificar o seu impacto no resultado da clusterização. Nessa avaliação, tanto o parâmetro ε quanto o parâmetro η foram sendo variados e constatou-se que o parâmetro ε possui uma influência considerável no resultado final do algoritmo, pois à medida que o raio de ε-vizinhança vai sendo aumentado algoritmo tende a formar *clusters* menos densos e mais dispersos. O parâmetro η também apresentou influência no resultado final, porém menor se comparada a ε.

Pelos testes realizados, conclui-se que os parâmetros de entrada do algoritmo também devem ser definidos com precisão, pois valores ligeiramente diferentes para esses parâmetros tendem a gerar resultados finais distintos.

5.2.2. Tempos de Processamento do DBSCAN na *Shell Orion*

Para obtenção de conjuntos com diversos tamanhos a base de dados original foi sendo gradativamente replicada, utilizando-se para isso o modelo de crescimento geométrico que torna possível a obtenção da visão do comportamento de diferentes tamanhos de cargas (centenas a milhares de registros) em um pequeno intervalo de iterações:

$$C_K = N * 2^{k-1}$$

Na Tabela 2 são verificados os tempos de processamento obtidos pelo algoritmo DBSCAN com relação ao tamanho da base de dados e as medidas de distância, concluindo-se que o tempo tem um crescimento aproximadamente linear com relação a quantidade de objetos.

Tabela 2 - Tempos de processamento do DBSCAN

Quantidade de registros	Distância Euclidiana	Distância Euclidiana Normalizada	Distância *Manhattan*
679	00m: 00s: 047ms	00m: 00s: 106ms	00m: 00s: 030ms
1358	00m: 00s: 187ms	00m: 00s: 335ms	00m: 00s: 080ms
2716	00m: 00s: 686ms	00m: 01s: 188ms	00m: 00s: 280ms
5432	00m: 02s: 605ms	00m: 04s: 754ms	00m: 01s: 010ms
10864	00m: 10s: 437ms	00m: 18s: 925ms	00m: 03s: 870ms
21728	00m: 41s: 309ms	01m: 16s: 311ms	00m: 16s: 160ms
43456	02m: 58s: 043ms	05m: 43s: 463ms	01m: 19s: 709ms
86912	13m: 01s: 347ms	21m: 37s: 672ms	05m: 41s: 684ms

5.2.3. Comparação com a ferramenta *Weka* 3.6

O algoritmo desenvolvido na *Shell Orion* foi comparado com o DBSCAN disponibilizado na ferramenta *Weka* 3.6 (http://www.cs.waikato.ac.nz/ml/weka/). Os atributos de entrada utilizados foram pH e vazão da água(l/s). A Tabela 3 exemplifica o resultado encontrado nas duas ferramentas.

Tabela 3. Testes de comparação entre o DBSCAN na Shell Orion e na *Weka* 3.6

Ferramenta	***Cluster* 1**	***Cluster* 2**	**Quantidade de *outliers***
Shell Orion	128 (18,85%)	531 (78,20%)	20 (2,95%)
Weka 3.6	128 (18,85%)	531 (78,20%)	20 (2,95%)

Observando-se que os resultados encontrados pela *Shell Orion* foram idênticos aos da ferramenta *Weka*, pode-se chegar à conclusão que em termos de resultados obtidos pelo DBSCAN, a *Shell Orion* implementa de maneira correta o algoritmo.

Para verificação da média de tempo de processamento necessário ao DBSCAN para realizar a tarefa de clusterização, tanto o algoritmo na desenvolvido na *Shell Orion*, quanto o algoritmo disponível na *Weka* foram executados um número fixo de vezes com os mesmos parâmetros de entrada.

A carga de dados foi definida replicando-se a base de dados de monitoramento de indicadores ambientais, utilizando-se para isso o modelo de crescimento geométrico com 4 iterações (5432 registros). O algoritmo DBSCAN foi executado 400 vezes cada ferramenta, totalizando 800 observações.

Em cada execução, o tempo de processamento do algoritmo foi coletado. A análise dos dados foi realizada com o auxílio do pacote estatístico *Statistical Package for Social Science*s (SPSS).

Inicialmente os dados coletados foram tabulados e organizados a fim de permitir a sua análise. Após isso, utilizou-se o gráfico *boxplot* com a finalidade de identificação de *outliers* entre as amostras, com a finalidade de evitar distorções na análise dos tempos. Após a identificação e exclusão dos *outliers*, os dados foram submetidos ao teste de aderência de *Kolmogorov-Smirnov* com a finalidade de verificar a hipótese de normalidade na distribuição das amostras.

O teste de *Kolmogorov-Smirnov* indicou a ausência de normalidade na distribuição de valores, o que torna proibitiva a utilização de testes paramétricos, como o teste *T* de *Student*, pois esse tipo de teste tem como um das suas pressuposições uma distribuição normal de valores em análise [BARBETTA; REIS; BORNIA, 2010].

Dessa maneira, optou-se por utilizar um teste não-paramétrico, visto que as suposições para aplicações de testes desse tipo são menos rígidas, o que possibilita uma aplicação mais generalizada. O teste não paramétrico de *U* de *Mann-Whitney* foi escolhido para comparação das médias de tempo de processamento.

O teste resultou em um *p-value* menor que o nível de significância pré-estabelecido (5%), portanto, com 95% de confiança, rejeita-se a hipótese nula e se aceita a hipótese alternativa que diz que as médias de tempo de processamento são estatisticamente diferentes (Tabela 4).

Tabela 4. Teste *U* de *Mann-Whitney*

Ferramenta	**Média (seg)**	**Mínimo (seg)**	**Máximo (seg)**	***p-value***
Weka 3.6	00m: 04s. 701ms	00m: 04s. 290ms	00m: 05s. 350ms	0.000
Shell Orion	00m: 04s. 558ms	00m: 04s. 480ms	00m: 04s. 680ms	

Pelas evidências estatísticas encontradas, tais como a média (00m: 04s: 701ms para a *Weka* e 00m: 04s: 558ms para a *Shell Orion*) e a mediana (00m: 04s: 660ms para a *Weka* e 00m: 04s: 560ms para a *Orion*) e pelo resultado do teste *U* de *Mann-Whitney* (Tabela 4), é possível afirmar que o algoritmo DBSCAN apresentou menor tempo de processamento na *Shell Orion Data Mining Engine.*

6. Considerações Finais

Este artigo apresentou o algoritmo DBSCAN que utiliza o conceito de clusterização baseada em densidade para a tarefa de clusterização da *Shell Orion Data Mining Engine*, contribuindo com a ampliação das funcionalidades da ferramenta.

Diante dos resultados obtidos pode-se confirmar a aplicabilidade do algoritmo para a tarefa de clusterização. A possibilidade de encontrar *outliers* entre os dados e a formação de *clusters* com formatos arbitrários, tornam o algoritmo uma opção interessante na clusterização de bases de dados com essas características. Também não existe a necessidade de ser informada a quantidade de c*lusters* desejados, o que faz com que o algoritmo imponha uma estruturação menos rígida aos dados.

Concluiu-se que o algoritmo foi desenvolvido com sucesso, pois apresentou funcionamento correto e resultados satisfatórios tanto na clusterização quanto nos resultados obtidos por meio da comparação com uma ferramenta de *data mining* conceituada e também com relação aos tempos de processamento.

Referências

ALEXANDRE, Nadja Zim; KREBS, Antonio Silvio Jornada. Discussão da aplicação do método do IQA na avaliação da qualidade das águas da região carbonífera de Santa Catarina. **Revista Tecnologia e Ambiente**, Criciúma, SC, v. 2, n. 1, p. 31-52, jun. 1996.

BARBETTA, Pedro Alberto; REIS, Marcelo Menezes; BORNIA, Antonio Cezar. **Estatística**: para cursos de engenharia e informática. 3. ed. São Paulo: Atlas, 2010.

BEZDEK, James et al. **Fuzzy models and algorithms for pattern recognition and image processing**. New York: Springer, 2005.

ESTER, Martin; KRIEGEL, Hans-Peter; SANDER, Jörg; XU, Xiaowei. A density-based algorithm for discovering clusters in large spatial databases with noise. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING (KDD-96), 2., 1996, Portland. **Proceedings…** . Portland: AAAI Press, 1996 p. 226-231.

FAYYAD, Usama M.; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. From data mining to knowledge discovery in databases**. AI Magazine**, Providence, v.17, n. 3, p. 37-54, autumn 1996.

HAN, Jiawei. KAMBER, Micheline. **Data mining**: concepts and techniques. 2. ed. San Francisco: Morgan Kaufmann, 2006.

KRIEGEL, Hans-Peter et al. Density-based clustering. In: PEDRYCZ, Witold (Org.). **Wiley interdisciplinary reviews**: data mining and knowledge discovery, New York, Wiley, 2011. p. 231-240.

MILLIGAN, Glenn W.; COOPER, Martha C. An examination of procedures for determining the Number of clusters in a data set. **Psychometrika**, Colombus, p. 159-179. Jun. 1985.

WITTEN, Ian H.; FRANK, Eibe; HALL, Mark A. **Data mining practical machine learning tools and techniques**. 3. ed. Burlington: Morgan Kaufmann, 2011.