DATA MINING: DETERMINAÇÃO DE AGRUPAMENTOS EM GRANDES BASES DE DADOS

Marcos Antonio de Almeida

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientador: Adilson Elias Xavier

Rio de Janeiro
Dezembro de 2013

DATA MINING: DETERMINAÇÃO DE AGRUPAMENTOS EM GRANDES BASES DE DADOS

Marcos Antonio de Almeida

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Adilson Elias Xavier, D.Sc.

Prof. Edmundo Albuquerque de Souza e Silva, Ph.D.

Prof. Luiz Fernando Loureiro Legey, Ph.D.

Prof. Ricardo Saraiva de Camargo, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2013

Almeida, Marcos Antonio de

Data Mining: Determinação de Agrupamentos em Grandes Bases de Dados/Marcos Antonio de Almeida. – Rio de Janeiro: UFRJ/COPPE, 2013.

xii , 78 p.: il ; 29, 7cm.

Orientador: Adilson Elias Xavier

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2013.

Referências Bibliográficas: p. 76 – 77.

1. Data Mining. 2. Clustering. 3. Big Data. I. Xavier, Adilson Elias. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Dedico este trabalho a minha
mãe, meu pai, minha esposa ,
meus filhos, neta e genro.

Agradecimentos

A Deus, que me dá saúde e forças para continuar lutando.

Ao meu orientador, Prof. Adilson Xavier, pelo estímulo, atenção e a incomensurável paciência generosamente dedicada durante a revisão e execução deste trabalho.

Ao Programa de Incentivo à Pós-Graduação da DATAPREV, pela oportunidade de realizar este trabalho.

Aos meus familiares pelo apoio, compreensão e colaboração.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

DATA MINING: DETERMINAÇÃO DE AGRUPAMENTOS EM GRANDES BASES DE DADOS

Marcos Antonio de Almeida

Dezembro/2013

Orientador: Adilson Elias Xavier

Programa: Engenharia de Sistemas e Computação

A explosão da taxa de crescimento do volume dos dados experimentada nas últimas décadas cria um novo desafio para os algoritmos de *Data Mining*. O objetivo desta dissertação é analisar o impacto deste fenomeno sobre as metodologias atuais e apresentar algumas abordagens emergentes para solução deste problema no campo de *clustering*.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

CLUSTERING FOR DATA MINING IN LARGE DATABASES

Marcos Antonio de Almeida

December/2013

Advisor: Adilson Elias Xavier

Department: Systems Engineering and Computer Science

The data exponential growth in last decade, creates a new challenge for Data Mining algorithms. The aim of this dissertation is analysing the impact of this phenomenon on current methodologies and present emerging approaches for solving this problem.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

1 Introdução

As últimas décadas têm sido pródigas em diversas áreas de aplicação da Tecnologia da Informação e Comunicação (TIC), mas este fato é particularmente notável em dois campos: Na captação e armazenamento de dados e nas redes de comunicação.

Tanto na esfera científica como no ambiente de negócios e até em nossa vida pessoal temos capturado, compartilhado e armazenado dados de forma cada vez mais intensa. O fenômeno da disseminação destas Base de Dados Gigantes pelos mais diversos campos de aplicação tornou-se um destacado tema de discussão atual : *Big Data*

Han J., na edição publicada em 2011 do seu livro sobre Mineração de Dados (*Data Mining*) [16], aborda este assunto afirmando que vivemos hoje na *era dos dados* e que para entrarmos de fato na *era da informação* é crítico que sejamos capazes de transformar dados em informação. Esta é a missão do *Data Mining*.

A presente dissertação de mestrado tem por objetivo analisar o impacto dessa explosiva taxa de crescimento do volume dos dados sobre as metodologias de *Data Mining* e apresentar algumas abordagens emergentes para solução deste problema no campo de *clustering*.

1.1 Mineração de Dados

Mineração de Dados (*Data Mining*) é uma área de pesquisa focada no desenvolvimento e aprimoramento de métodos computacionais para descoberta de padrões de interesse numa Base de Dados. Mas o que é um padrão interessante?

Um padrão para ser interessante deve ser válido, compreensível por quem vai usá-lo e de valor para o cliente. Este último requisito varia muito em função do contexto de onde o *Data Mining* está sendo aplicado.

Por exemplo, os *outliers* são observações muito afastadas das demais observações presentes numa base de dados. Na maioria dos casos estas observações devem ser expurgadas da base de

dados para não afetar os resultados. Entretanto, no caso de uma aplicação para detecção de fraude ou de intrusão na rede, passam a ser o verdadeiro foco do interesse.

Com relação ao primeiro requisito, como garantir que o padrão é válido? Fayyad, Shapiro e Smyth afirmam em seu artigo [12] iniciam a resposta a esta questão lembrando que o termo *Data Mining* surgiu no meio estatístico com uma conatação negativa.

Isto porque é possível extrair padrões que parecem ser estatísticamente significativos de qualquer base de dados suficientemente grande. Por esse motivo, todo algoritmo tem seu critério de avaliação da conformidade da aderência do resultado ao modelo utilizado.

A pesquisa em *Data Mining* ganhou impulso nos anos 80, do século passado, devido à crescente dificuldade em se analisar os dados armazenados nas empresas e no meio científico com os métodos tradicionais.

Para enfrentar este desafio foram combinadas abordagens originárias de diversas áreas do conhecimento. Han [16], por exemplo, elenca as seguintes áreas : Estatística, Aprendizado de Máquina, Reconhecimento de Padrões, Banco de Dados, Visualização, Algoritmos, Recuperação da Informação, Computação de Alto Desempenho. A forte ligação entre *Data Mining* e a Estatística é evidente quando se recorda que o trabalho do estatístico consiste em: Coletar, Analisar, Interpretar e Exibir Dados. Seu objetivo é fazer inferências (predições) sobre o futuro ou trazer explicações sobre os dados analisados.

A fundamentação dos modelos estatísticos está na probabilidade matemática e sua principal aplicação é no estudo de fenômenos aleatórios. Testes de hipóteses, significância estatística e estimativas de erros estão entre as contribuições para o *Data Mining*.

O campo de Aprendizado de Máquina também proporcionou importantes contribuições para o *Data Mining*. A pesquisa nesta área investiga meios de fazer as máquinas (computadores) aprenderem. Sua contribuição para o desenvolvimento do *Data Mining* tem sido imensa.

Por exemplo, nas técnicas de Aprendizado Supervisionado, nas quais um especilista classifica previamente os objetos de uma base de dados de treinamento têm grande aplicação nos problemas de classificação e regressão.

O Aprendizado não Supervisionado é outra área do aprendizado de máquina na qual ocorreram importantes contribuições. O principal problema desta área é a análise de agrupamentos ou *clustering*, que busca encontrar a melhor maneira de agrupar um dado conjunto de objetos.

Estes e outros importantes problemas de *Data Mining* terão seu algoritmos abordados sucintamente no capítulo de revisão da literatura.

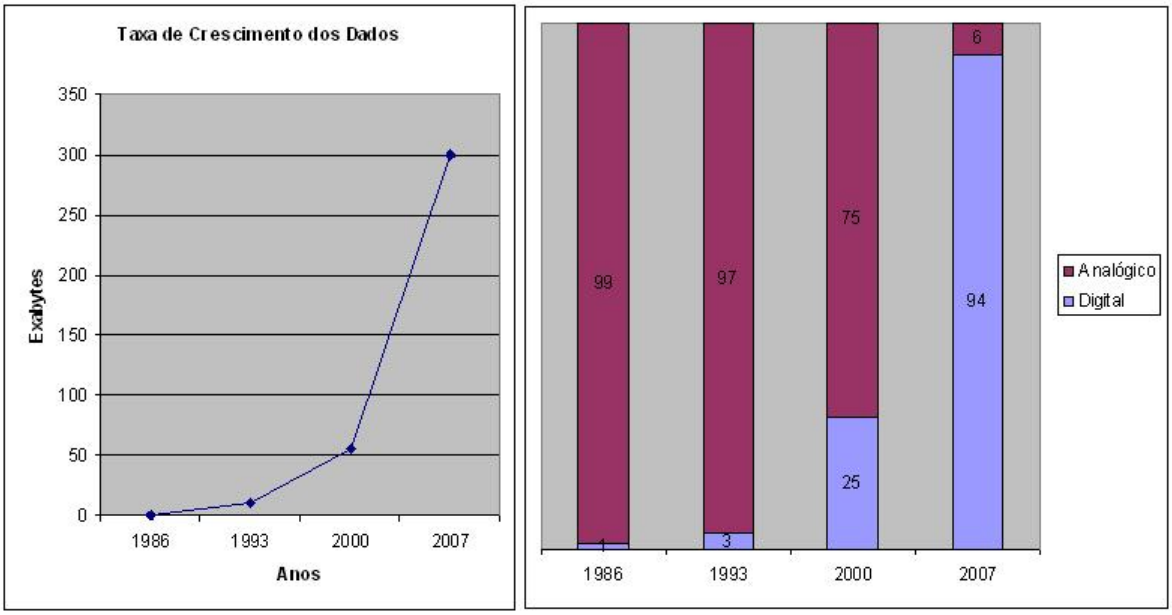

Figura 1.1: Fonte: Hilbert et Lopez , Science , 2011

1.2 Bases de Dados Gigantes

O crescimento exponencial experimentado pela aquisição de dados durante todo este tempo, está sendo chamado de *Big Data* e sendo considerado como uma nova era.

O gráfico 1.1 apresentado por Hibert em 2012 [20] mostra que houve uma forte inflexão na taxa de crescimento do armazenamento de dados no ano 2000. Mostra ainda uma massiva substituição dos meios aquisição e armazenamento de dados analógicos por digitais.

Na ciência este fenômeno foi disparado pelo projeto do genoma humano na genética, nos novos telescópios na astronomia, nas imagens de satélite, na física das partículas, etc., de forma tão abrangente que foi apresentado como um novo paradigma , o 4o paradigma.

Segundo este paradigma, o cientista passa a lidar apenas com os dados do experimento, sem a necessidade de lidar diretamente com os instrumentos de medida.

O título desta dissertação em ingles é quase o mesmo de um artigo publicado em 1998 por Xu et all *Clustering for Data Mining in Large Databases* [29] . Esta escolha foi feita para provocar a seguinte reflexão: estamos entrando em uma nova era do exabyte?.

No artigo, os autores expressam sua preocupação com o volume de dados gerados pelo projeto da Agencia Espacial Norte Americana (NASA) de colocar em órbita um satélite (Terra) com um telescópio de observação terrestre, capaz de gerar imagens na taxa de 50 gigabytes de informação por hora.

Se toda capacidade de transmissão foi utilizada até hoje, passados 15 anos, foram armazendos 8.5 pentabytes de dados de imagem somente deste satélite.

No campo social este fenômeno manifestou-se em sua plenitude com o advento das redes sociais. Publicações de mensagens e fotos nos micro-blogs e sites de relacionamento concorrem para um fluxo ininterrupto de dados disponíveis para quem souber interpretá-los.

As empresas de internet tiveram um papel preponderante neste fenômeno, pois, ao mesmo tempo em que ofereceram as condições para que este fenômeno se difundisse pela webesfera, também foram as que sofreram primeiro as consequências desta avalanche de dados.

Como prêmio, aquelas que conseguiram acompanhar este rítmo conseguiram se posicionar fortemente nos seus mercados. *Google* , *Facebook* e *Amazon* são exemplos recorrentes, precisaram desenvolver tecnologias para enfrentar estes desafios.

A Google, por exemplo, desenvolveu uma arquitetura de clusters [3] com computadores de baixo custo (*comodities*) e avançadas técnicas de processamento intensivamente paralelo (*Massive Parallel Processing*) [9]

Além das empresas de internet , as empresas de telecomunicações, varejo, finanças, entretenimento, já têm seus negócios fortemente influênciado por este volume de dados, fato que abre uma grande oportunidade para empresas de Tecnologia da Informação e Comunicação (TIC), notadamente na área de infra estrutura de TIC, seja pela demanda de seus usuários, ou pelas oportunidades/riscos que surgem no mercado.

1.3 Dataprev

A DATAPREV, como o braço de Tecnologia da Informação e Comunicação (TIC) da Previdência Social, cumpre um importante papel na prestação de serviços de qualidade ao cidadão. Cumprir este papel exige a contínua qualificação de seu corpo funcional.

Não por um acaso, uma das áreas onde existem grandes oportunidades e desafios para serem enfrentados no âmbito da DATAPREV é no *Data Mining* em Grandes Base de Dados.

Um artigo de 2006 de Davenport *Competing in analitcs* [8] , chama atenção para o êxito das empresas que estavam obtendo sucesso pela habilidade de colecionar, analisar e agir, baseado nos dados.

Uma empresa como a Dataprev, fiel depositária de uma das maiores base de dados sociais do Brasil, tem a oportunidade de oferecer aos seus clientes, sofisticados instrumentos analíticos para tomada de decisão, promovendo assim uma contribuição para a melhora na prestação de serviços ao cidadão.

Graças ao Programa de Incentivo à Pós-Graduação da Empresa foram criadas oportunidades para o desenvolvimento do presente trabalho. O tema desta dissertação: *Data Mining* em Grandes Bases de Dados, foi escolhido para se alinhar ao planejamento estratégico da DATAPREV.

Uma clara demonstração do acerto desta escolha foi a utilização dos conhecimentos adquiridos durante as pesquisas para realização deste trabalho nas soluções técnicas do Projeto *BigData* Perícias Médicas , projeto piloto inovador no contexto da Empresa.

2 *Revisão da Literatura*

2.1 Principais Algoritmos de *Data Mining*

Esta revisão dos algoritmos de *Data Mining* se baseia na lista dos dez mais influentes algoritmos desta área [25] e no livro homônimo publicado em 2009 [27]. Os algoritmos consagrados no IEEE-ICDM (*International Conference on Data Mining*) de Dezembro de 2006 são : C4.5, k-means, SVM, Apriori, EM, Page Rank, Ada Boost, kNN, Naive Bayes e CART.

	Algoritmo	Campo de Pesquisa	Autor Artigo de Referência	Ano	citações
1	C4.5	Classificação	Quinlan, J. R.	1993	6907
2	K-Means	Clustering	MacQueen, J. B.	1967	1579
3	SVM	Aprendizagem Estatística	Vapnik, V. N.	1995	6441
4	APRIORI	Regras de Associação	Agrawal,R. e Srikant, R.	1994	3639
5	EM	Aprendizagem Estatística	McLachlan, G. e Peel, D.	2000	848
6	Page Rank	Análise de Links	Brin, S. and Page, L.	1998	2558
7	kNN	Classificação	Hastie, T. e Tibshirani, R.	1996	183
7	Naive Bayes	Classificação	Hand, D.J., Yu, K.,	2001	51
7	AdaBoost	Bagging and Boosting	Freund, Y. e Schapire, R. E.	1997	1576
10	CART	Classificação	Breiman, L. et all	1984	6078
-	HITS	Análise de Links	Kleinberg, J. M.	1998	2240
-	FP-Tree	Regras de Associação	Han, J. et all	2000	1258
-	BIRCH	Clustering	Zhang, T. et all	1996	853
-	GSP	Padrões de Sequencias	Srikant, R. and Agrawal, R.	1996	596
-	CBA	Integrated Mining	Liu, B. ett all	2002	436
-	Finding reduct	Rough Sets	Zdzislaw Pawlak	1992	329
-	PrefixSpan	Sequential Patterns	Pei, J. et all	2001	248
-	gSpan	Graph Mining	Yan, X. e Han, J.	2002	155

Tabela 2.1: Finalistas ao Top 10 Algoritimos em *Data Mining*

Esta lista foi elaborada a partir de uma enquete (survey) junto aos ganhadores dos principais prêmios dos dois mais importantes congressos de Data Mining : ACM-KDD e IEEE-ICDM. Este grupo de pesquisadores foi convidado a responder quais eram os 10 mais influentes algoritmos no seu campo de pesquisa e de outras áreas do *Data Mining*, o motivo de sua escolha e

a indicação de um artigo de referência sobre o algoritmo.

O conjunto obtido foi filtrado para manter apenas aqueles cuja referência indicada atingisse ao menos 50 citações no *Google Scholars*, em outubro de 2006. Com este critério, restaram apenas 18 algoritmos.

Assim, no congresso IEEE-ICDM de 2006 os algoritmos finalistas foram submetidos a votação pelos congressistas presentes. A tabela 2.1 apresenta a lista desses finalistas.

Segundo o survey, os campos de pesquisa nos quais estes algoritmos são amplamente utilizados compreendem : Classificação, *Clustering* , Aprendizado Estatístico, Regras de Associação e Análise de link.

Campo	% Citações	Votos	% Votos	Algoritimos
Classificação	37%	185	37%	4
Aprendizado Estatístico	20%	106	21%	2
Regras de Associação	14%	52	11%	2
Análise de Link	13%	46	9%	2
Clustering	7%	60	12%	2
Bagging and Boosting	4%	45	9%	1
Demais	4%	-	-	5

Tabela 2.2: Top 10 por Campo de Pesquisa

A Tabela 2.2 mostra as citações registradas no Google Scholars, bem como o resultado da votação apurada entre os congressistas presentes no IEEE-ICDM 2006. Como mostra a primeira coluna desta tabela, os quatro primeiros campos de pesquisa respondem por 84% das citações : Classificação (37%), Aprendizado Estatístico (20%), Regras de Associação (14%) e Análise de Link (13%).

Note-se ainda que apesar de *K-Means* aparecer em 2o lugar na votação final, a participação de *Clustering*(7%) nas citações é menor do que Analise de Link (13%). Observa-se finalmente que os campos *Graph Mining* , *Integrated Mining*, *Rough Sets* e *Sequential Patterns* não conseguiram classificar nenhum algoritmo entre os dez finalistas.

Quando se analisa o resultado das citações dos algoritmos individualmente, nota-se, por exemplo, que kNN (1%) e Naive Bayes (menos de 1%) ficaram entre os top 10 em detrimento de Hits(6%) e FP-Tree(3%) que tiveram maior número de citações. Isto aconteceu porque a ultima etapa da eleição dos top 10 foi feita levando-se em consideração unicamente os votos apurados no IEEE-ICDM 2006.

Para descrever cada um destes algoritmos, optamos por organizá-los por campo de pesquisa.

Começando com os algoritmos de aprendizado supervisionado, iniciaremos com os classificadores, ou seja, os algoritmos que descobrem a classe a que um objeto pertence: CART, C4.5, kNN, Naive Bayes. Depois iremos abordar os algoritmos de aprendizado estatístico: SVM e EM e, por fim, os demais: AdaBoost, Apriori.

Terminaremos esta apresentação com os algoritmos do tipo aprendizado não supervisionado *k-means*, um algoritmo de *clustering* e o Page Rank. .

Esperamos entregar, como resultado deste resumo, uma visão unificada destes algoritmos, tendo como base o trabalho elaborado por dez especialistas em seus respectivos campos de pesquisa.

2.1.1 Classificadores : CART, C4.5 , kNN , Naive Bayes

Um classificador é um algoritmo de Aprendizado Supervisionado de Máquina e, como tal, usa a informação contida nos Dados de Treinamento, para descobrir regras que permitam decidir a que classe uma nova entrada pertence.

Estes algoritmos processam tipicamente em duas fases: Uma de aprendizagem e outra para treinamento.

A fase de aprendizagem tem por objetivo descobrir critérios que devam compor as regras de classificação, que são as regras que decidem a que classe uma observação pertence.

Na fase de treinamento, busca-se verificar se as regras formuladas na fase anterior atingem a resultados satisfatórios. Para isso, aplicam-se as regras obtidas na aprendizagem sobre os dados de treinamento e compara-se o resultado com o esperado.

Isto é possível porque os Dados de Treinamento são um conjunto de objetos previamente preparados por um supervisor ou especialista com a indicação da classe a que cada objeto pertence.

Uma técnica muito utilizada é o *cross-validation* que consiste na divisão dos Dados de Treinamento em duas partes: uma para ser usada na aprendizagem e outra no treinamento.

A regra obtida na fase de aprendizagem pode ser expressa de diversas formas. Veremos primeiro os algoritmos que, para isto, constroem uma árvore de decisão: CART e C4.5. Em seguida, kNN, que usa critérios de proximidade e, por fim, Naive Bayes, que adota critérios de vizinhança.

2.1.2 CART

Segundo Dan Steinberg, autor do Capítulo 10 sobre CART no livro [27], o artigo *Classification and Regression Trees* de L. Breiman, J. Friedman, R. Olshen, and C. Stone de 1983 [5], teve grande impacto no desenvolvimento das áreas de Inteligencia Artificial, Aprendizado de Máquina, Estatística não Paramétrica e no Data Mining. Isto porque, o artigo lança novas bases para o estudo e uso de árvores de decisão em problemas de classificação.

Em seu artigo, o autor admite que a maioria dos problemas estatísticos são paramétricos, isto é, problemas nos quais a população estudada apresenta uma distribuição que segue uma função de densidade de distribuição expressa por um conjunto de parâmetros.

No caso do modelo gausiano, por exemplo, a distribuição normal $\eta(\mu,\sigma)$ é função dos parâmetros μ (média) e σ (desvio padrão).

No entanto, o autor alerta que existem problemas de *Data Mining* nos quais a base de dados não se encaixa em nenhum dos modelos estatisticos conhecidos e/ou não apresenta uma distribuição que seja governada por algum conjunto de parâmetros.

Para estes casos existem os metodos estatísticos não paramétricos. O CART *Classification And Regression Trees* é um desses métodos.

O objetivo do algoritmo CART é criar uma Árvore Binária de Decisão (sim/não). Esta árvore será utilizada para classificar as novas observações, ou seja, para decidir a que classe esta observação pertence. Isto se consegue percorrendo os nós da árvore, do topo até um nó folha, onde se encontrará a indicação da classe a que esta observação pertence.

Durante o seu trajeto em direção ao nó folha, a pesquisa passa por diversos nós. Em cada um destes nós encontra-se uma regra que decide se o próximo nó a ser visitado está à direta ou à esquerda. Descobrir as regras que devem ser usadas em cada nó, de modo a construir a árvore ótima, é o desafio que o CART se propõe a resolver.

Como todo classificador, o CART usa uma parte dos Dados de Treinamento, para criar as regras e a outra parte, para testar seus resultados.

A construção desta árvore é processada em duas fases: Ramificação e Poda. Na fase de ramificação, a árvore expande suas ramificações (*tree-growing*) gerando a árvore de tamanho máximo e na fase de poda, suas folhas são aparadas(podadas) (*pruning*) seguindo um critério de desempenho no teste. No final destas fases obtem-se a árvore ótima.

Fase 1: Ramificação

Algoritmo 2.1.1: CART - Fase 1:Ramificação

```
Data: Dados de Treinamento
Result: Árvore Máxima
initialization;
Aloca todos dados de treinamento ao nó raiz;
Define raiz como folha;
while houver folhas que devam ser processadas na lista do
    carregue folha;
    testa se a folha deve ser dividida;
    if folha deve ser dividida then
        decubra o atributo que melhor divide a folha;
        divida a folha em duas;
        inclua as folhas na lista;
    else
        retira a folha da lista ;
```

Na inicialização da Fase de Ramificação o algoritmo aloca todos os dados de treinamento no nó raiz , define-o como folha e inicializa com o nó raiz, a lista de folhas a serem processadas.

O passo seguinte é iterar pela lista de folhas a serem processadas testando se estas devem ser divididas. A decisão de dividir a folha é feita com base em dois testes. No primeiro teste, adota-se algum critério de tamanho da folha. No segundo teste, verifica-se se todos os elementos presentes na folha são da mesma classe. Basta que um dos testes seja positivo, para que a folha saia das lista de folhas a serem divididas.

Quando o teste decide dividir, o módulo de divisão (*split*) busca entre os atributos aquele que melhor divide o conjunto em duas partições. Expressa esta regra de uma forma simples, *atributo* $X_i \leq C$. Assim, cada elemento da folha é avaliado segundo este critério e vai para a folha da esquerda, se atender a regra e para a folha da direita, caso contrario.

Para descobrir qual é o melhor atributo que divide a folha , os autores adotaram o indice de GINI que para um nó t é dado pela expressão:

$$G(t) = 1 - p(t)^2 - (1 - p(t))^2 \tag{2.1}$$

onde $p(t)$ é a frequencia relativa da classe 1 no nó t (que pode ser ponderada por um vetor **w**)

Define-se ainda um índice ganho (*improvement*) gerado pela divisão de uma folha **P** , aplicando-se a formula :

$$I(P) = G(P) - qG(L) - (1-q)G(R) \tag{2.2}$$

onde $G(L)$ e $G(R)$ são os nós esquerdo e direito que serão criados e **q** é um vetor de ponderação.

Finalmente, toma a decisão baseado no índice:

$$I(split) = [.25(q(1-q))^u \sum_k |P_L(k) - P_R(k)|]^2 \tag{2.3}$$

onde k é o indice da classe k, P_L e P_R são as probabilidade de distribuição da classe k nos nós esquerdo e direito. A potencia u e o fator q são parâmetros de controle do peso da penalidade de se fazer uma divisão (split), ajustáveis pelo usuário.

Fase 2: Poda da Árvore

Algoritmo 2.1.2: CART - Fase 2: Poda da Árvore

Data: Árvore Máxima gerada na Fase 1

Result: Árvore de Classificação Otimizada

t_{Left} = ponteiro nó esquerdo;

t_{Right} = ponteiro nó direito;

$r(t)$ = taxa de mistura de classes no nó t;

$p(t)$ = percentual dos dados no nó t;

$R(t) = r(t) * p(t)$;

$|T|$ numero de folhas;

tMax= árvore máxima;

tCurrent= tMax;

forall *nós pais t de duas folhas* **do**
  Remova todos nós pais nos quais $R(t) = R(t_{Left}) + R(t_{Right})$

$Current_{tree} = tMax$ após Poda ;

if $|Current_{tree}| = 1$ **then**
  termina;

else
  forall *nós pais t de duas folhas* **do**
    Remova todos nós pais nos quais $R(t) = R(t_{Left}) - R(t_{Right})$ é minimo.
  $Current_{tree} = tMax$ após poda ;

Na fase de poda, o objetivo do algoritmo é obter a árvore ótima para classificação. Vamos considerar aqui a versão apresentada por Steinberg no Capitulo 10 do livro [27].

O algoritmo parte da árvore máxima gerada na fase de ramificação e remove todas ramificações que não melhoram a precisão atingida durante os testes com os dados de treinamento. O processo de poda prossegue removendo todos os nós que não contribuam favoravelmente para o desempenho do teste. Esta poda envolve a remoção das folhas para o nó pai que passa a formar uma única folha com seus filhos.

O algoritmo 2.2.2 apresenta a versão simplificada publicada no Capitulo 10 do livro [27], na qual só se consideram os pais das folhas terminais da árvore.

Vantagens , Desvantagens e Desafios Atuais

Em relação às escolhas feitas, os autores do CART [5] argumentam que é preferível o uso de árvores binárias, pela menor taxa de fragmentação, do que árvores com mais de dois ramos. Além do mais, é permitido o uso do mesmo atributo nos níveis subsequentes.

Argumentam ainda que, qualquer perda de clareza na leitura das regras montadas na árvore é compensada pelo ganho de performance. Entretanto, nesta escolha, pesou também o fato de que a fundamentação teórica é baseada no fracionamento binário.

Segundo Steinberg [27], uma caracteristica importante do CART, em comparação com os outros algoritmos (C 4.5 por exemplo), é que o método não usa uma métrica teórica de performance para seleção da árvore. Sua performance é medida exclusivamente por um teste de avaliação da árvore nos Dados de Teste ou via validação cruzada (cross-validation).

O mecanismo do CART também pode incluir opcionalmente um balanceamento automático de classes, assim como um tratamento automático de valores ausentes e ruídos (*outliers*).

Além disso, o CART consegue resolver problemas de aprendizado com tomada de decisão sensível ao custo, ou seja, que toma decisão levando em consideração o efeito (custo da decisão).Um exemplo deste caso encontramos no diagnóstico de uma doença, onde o falso negativo pode ter um custo maior que o falso positivo, pois um erro deste tipo pode levar à morte do paciente.

CART lida bem com dados de alta dimensionalidade pois consegue gerar um resultado usando poucas variaveis.

Uma importante fraqueza é que não há nenhum intervalo de confiança ou probabilidade de acerto para a classificação prevista para um novo dado, usando a árvore de decisão gerada pelo algoritimo, uma vez que o CART não é baseado num modelo probabilístico.

Podem ocorrer árvores de decisão instáveis, onde pequenas modificações nos dados de treinamento podem produzir mudanças radicais, aumentando ou diminuindo a complexidade da sua estrutura.

2.1.3 C4.5

Introdução

O C4.5 foi o vencedor do eleição dos *Top Ten Algorithms in Data Mining*. O artigo de Quinlan, J. R., C4.5: Programs for Machine Learning [24], referência por excelência neste assunto, tinha o expressivo numero de 6907 citações no *Google Scholars*, em dezembro de 2006. Quinlan também é um dos autores do artigo sobre os *Top Ten* [25] que apresenta um resumo de cada um dos dez algoritmos eleitos.

O objetivo de um classificador, como o C4.5, é montar uma árvore de decisão para que seja capaz de decidir em que classe uma nova entrada pertence, a partir de um conjunto de dados com a indicação da classe a que cada objeto pertence (Dados de Treinamento).

Dessa forma, os algoritmos C4.5 e CART tem o mesmo objetivo. Por isso são bastante similares. Entretanto, existem algumas diferenças fundamentais. Quilan elenca as principais.

A primeira diferença é que, ao contrário do CART que usa árvores binárias, o C4.5 permite mais de duas ramificações em cada nó.

Além disso, CART usa o índice de GINI como indicador de pureza do nó enquanto o C4.5 usa um critério de ganho de informação.

Outra diferença apontada é que CART poda árvores e usa um modelo de custo apurado durante o teste, enquanto C4.5 usa um critério de limite de confiança.

Finalmente, Quilan [24] faz uma digressão sobre diferenças no método de tratar valores faltantes. CART usa um teste alternativo, substituindo o campo ausente por outro que apresente um resultado semelhante, enquanto o C4.5 faz um tramento probabilístico.

pseudo-codigo

Algoritmo 2.1.3: C4.5

Data: Dados de Treinamento D

Result: Árvore Otimizada

inicialização;

$arvore =$ Null;

Aloca todos dados de treinamento ao nó raiz;

if *D é "puro"OU outro criterio de parada* **then**
  Fim do processamanto;

forall *atributos* $a \in D$ **do**
  Calcule critério sobre a
$a_{melhor} =$ melhor atributo segundo critério;

$arvore =$ crie um nó de decisão usando o teste a_{melhor} ;

$D_v =$ sub-datasets baseados em a_{melhor} ;

forall $D_v =$ **do**
  $arvore_v = C4.5(D_v)$;
  anexa $arvore_v$ ao galho correspondente da árvore ;

Retorne Árvore ;

fase 1: Ramificação

Algumas estratégias são adotadas para escolher um teste. Por exemplo, se o atributo for numérico podemos usar a média ou mediana como critério para ramificar o nó. As melhores estratégias são baseadas no critério de ganho de informação que minimiza a desordem total ou no critério de taxa de divisão da informação, que é o mais usado.

fase 2: Poda da árvore

Em toda técnica de aprendizado de máquina é necessário dar uma folga, para que a regra não fique viciada na massa de treinamento.

O algoritmo de poda é baseado numa estimativa pessimista da taxa de erro associada aos casos do conjunto D que não pertençam a classe mais frequente do conjunto.

Vantagens e Desvantagens

Quilan [24] admite que quando as árvores de decisão ficam muito complexas sua compreensão fica comprometida. A alternativa, neste caso, é construir uma regra para cada classe. Algo do tipo, Se $regra_1, regra_2, \ldots, regra_n$ então classe é X . Entretanto, o autor alerta que seu uso requer quantidades consideráveis de CPU e memória para construi-las.

Algumas das questões em aberto para pesquisa mostram onde o C4.5 ainda tem desafios para superar. A construção de árvores estáveis e a decomposição de árvores complexas são alguns destes temas.

No primeiro caso, temos a situação de árvores que são muito afetadas com a inclusão de um caso que não pertencia aos Dados de Treinamento.

No segundo caso, o objetivo da pesquisa é decompor uma árvore complexa num conjunto de árvores mais simples que sejam equivalentes em termos de resultados preditivos.

Uma das vantagens desta estratégia, é facilitar a compreensão da árvore complexa usando árvores mais simples. Outra, é usar estas árvores para *boosting*, que é um dos algoritmos que serão abordados a seguir.

2.1.4 k-NN

"Diga com quem andas e direi quem és", é um provérvio popular que resume o princípio no qual o k-NN (*k Nearest Neighbors*) se baseia.

Este princípio da sabedoria popular pode ser expresso matemáticamente por:

$$y = \arg\max_{v} \sum_{z_i \in D_z} I(v = y_i) \tag{2.4}$$

onde D é a Base de Dados de Treinamento, ou seja, o conjunto de objetos $z = (x,y) \in D$, onde x é o vetor de atributos do objeto z e y a sua classe (*label*). Seja $D_z \subset D$ o conjunto formado pelos k objetos de treinamento mais próximos de um objeto de teste $z = (x,y)$.

Se a função identidade $I(.)$ for definida por:

$$I(v = y_i) = \begin{cases} 1, & \text{se} \quad v = y_i \\ 0, & \text{Caso} \quad \text{Contrário.} \end{cases}$$

Então , v será a classe com maior frequência na vizinhança de z.

Michael Steinbach e Pang-Ning Tan , autores do capítulo 8 do livro [27] sobre K-NN e dois dos autores do artigo sobre os *Top Ten* [25], elencam os principais aspectos do algoritmo.

Em primeiro lugar, citam a questão da métrica de similaridade e/ou distância usada para calcular a proximidade entre as observações.

Outro aspecto considerado relevante é a escolha do parâmetro k que define o número de objetos que uma vizinhança deverá conter.

Finalmente, indicam o critério de determinação da classe do objeto de teste a partir da classe dos vizinhos.

Assim, as decisões tomadas sobre qual métrica será escolhida, o número de vizinhos que irá compor a vizinhança e o critério de determinação utilizado para definir a classe , são de fundamental importância.

pseudo-código

Algoritmo 2.1.4: kNN k Nearest Neighbor Classification

Data: Dados de Treinamento D, objeto de teste z,conjunto classes usadas L

Result: $c_z \in L$ classe do objeto z

inicialização;

forall $y \in D$ **do**
  Compute $D(Z,y)$ distância entre z e y

Selecione $N \subset D$ dos k objetos mais próximos de z ;

Retorne $c_z = \arg\max_{v \in L} \sum_{y \in N} I(v = class(c_y))$;

Vantagens , Desvantagens e Pesquisas em Andamento

kNN é uma Máquina de Aprendizagem Preguiçosa, *lazy learners*, ou seja, ao contrário das Máquinas Aprendizagem Ávida, *eager learners*, não requer que se construa uma regra de classificação, como o C4.5, que constrói uma árvore de decisão. Esta simplicidade permite que o método se ajuste bem a problemas de classificação mais complexos, como quando um objeto pode pertencer a mais de uma classe.

Além disto, o algoritmo é fácil de entender e de implementar conseguindo bons resultados em diversas situações como, por exemplo, em sua aplicação na análise dos dados obtidos por equipamentos de *microarray*, usados para pesquisa genética.

Os autores, entretanto, alertam para algumas das desvantagens do método, como o da sensibilidade da métrica euclidiana ao número de atributos da base de dados. Neste caso, há uma perda da capacidade de medir a proximidade entre as observações, à medida que este número aumenta.

Outro ponto negativo é a necessidade de tratarmos a base de dados, antes de aplicar o algoritmo para prevenirmos que alguma das dimensões domine de forma preponderante o critério de proximidade, apenas por estar expresso numa ordem de grandeza numérica, maior que as demais.

Outro problema indicado pelos autores é tamanho da base de treinamento (que possui os objetos classificados). Na medida em que a base cresce sua eficácia diminui. Isto se deve à necessidade de se computar a distância de cada observação ao objeto de teste, para encontrarmos os k vizinhos mais próximos.

2.1.5 Naive Bayes

O objetivo de um classificador é descobrir um critério de classificação utilizando-se dos dados de treinamento para isso. O Naive Bayes consegue isto criando um simples vetor de pontuação (*score*), que define uma faixa de pontuação (*threshold*) para enquadramento de cada classe.

Ou seja, partindo de uma base de dados para treinamento D, formada por objetos $x \in D$, compostos por p atributos $x = (x_1, x_2, \ldots, x_p)$ e a indicação da classe i a que pertence , vamos construir uma pontuação baseada em qualquer função monotônica (ln por exemplo) de $P(i|x)$, cuja notação significa a probabilidade de i acontecer dado que x acontece , ou mais explícito ainda, de i acontecer quando $x_1, x_2, \ldots, x_p$ acontecem simultaneamente.

Se considerarmos que todos os p atributos de x são termos independentes, em relação à sua classe, então poderemos aplicar o teorema de Bayes, expresso por:

$$P(i|x) = P(x|i)P(i)/P(x) \tag{2.5}$$

Onde $P(i)$ e $P(x)$ são respectivamente a probabilidade *a priori* (teórica) dos eventos i e x ocorrerem e $P(i|x)$ é a probabilidade de i ocorrer *a posteriori* de x, ou seja, de i ocorrer uma vez que x ocorre.

Analogamente, $P(x|i)$ é a probabilidade de x *a posteriori* de i. Na expressão de Bayes, $P(x|i)$ é chamado de verossimilhança (*likelihood*) e $P(x)$ de evidência.

Esta restrição de independência da classe é tão importante para o método que o mesmo é chamado alternativamente de ”Bayes independente”.

Hand, D.J., autor do Capítulo 9 do livro [27] e também do artigo que representou o algoritmo, *Idiot's Bayes: Not So Stupid After All?*, constroi sua argumentação considerando apenas duas classes (classe 0 e classe 1).

Partindo da premissa que nada sabemos sobre a probabilidade *a priori* destas classes $P(0)$ e $P(1)$ e usando a função dada pela razão $P(1|x)/P(0|x)$ como a função distribuição condicional $f(i|x)$ de x para classe i, então a função $f(x)$, responsável pela pontuação de uma entrada x, é expressa pela equação:

$$f(x) = f(x|0)P(0) + f(x|1)P(1) \tag{2.6}$$

Utilizando a função $P(1|x)/P(0|x)$, monotônica em relação a distribuição condicional $f(i|x)$, temos que:

$$\frac{P(1|x)}{P(0|x)} = \frac{f(x|1)P(1)}{f(x|0)P(0)} \tag{2.7}$$

Como supomos que os componentes de $f(x_j|i)$ são independente da classe, podemos trabalhar com p distribuições de uma só varivável. Dessa forma, podemos reescrever $f(x|i)$:

$$\frac{P(1|x)}{P(0|x)} = \frac{\prod_{j=1}^{p} f(x_j|1)P(1)}{\prod_{j=1}^{p} f(x_j|0)P(0)} = \frac{P(1)}{P(0)} \prod_{j=1}^{p} \frac{f(x_j|1)}{f(x_j|0)} \tag{2.8}$$

$$\ln \frac{P(1|x)}{P(0|x)} = \ln \frac{P(1)}{P(0)} + \sum_{j=1}^{p} \ln \frac{f(x_j|1)}{f(x_j|0)} \tag{2.9}$$

fazendo $k = \ln \frac{P(1)}{P(0)}$ e $w_j = \ln \frac{f(x_j|1)}{f(x_j|0)}$ simplificamos para

$$\ln \frac{P(1|x)}{P(0|x)} = k + \sum_{j=1}^{p} w_j \tag{2.10}$$

pseudo-código

Uma das pricipais aplicações do Naive Bayes é na classificação de documentos e na detecção de *spam* em e-mails. O algoritmo 2.2.5 considera este caso.

Algoritmo 2.1.5: Naive Bayes Treinamento

Data: Dados de Treinamento D, Classes C

Result: $P(a_i|c_i)$

inicialização;

$V = vocabulos(D)$;

$n = count(V)$;

$t = 1$;

foreach *classe* $c_i \in C$ **do**

  $D_i \subset D$;

  $P(c_i) = |D_i|/|D|$;

  $n_i = cont(vocabulos(D_i))$;

  foreach *word* $a_j \in V$ **do**

    $P(a_i|c_i) = (n_{ij}+1)/(n_i+n)$

Vantagens e Desvantagens

Uma das vantagens do método é poder usar o algoritmo diretamente nos dados brutos, ou seja, sem a necessidade de tratamento prévio.

Esse algorimo também é fácil de construir, bem como de interpretar seus resultados. Além disso, geralmente tem um bom desempenho computacional.

O autor conclui o artigo argumentando que Naive Bayes pode não ser o melhor classificador em uma ou outra aplicação específica, mas é uma das primeiras opções de algoritmo para se iniciar um trabalho de classificação pois funciona bem em uma grande gama de aplicações.

2.1.6 Aprendizado Estatísticos: SVM e EM

A teoria de aprendizagem estatística, que nasce do intercambio entre a estatística e o aprendizado de máquina, tem no SVM (*Support Vector Machine*) e no EM (*Expectation-Maximization*) duas das suas mais relevantes contribuições aos algoritmos de *Data Mining*

2.1.7 SVM

O ponto de partida teórico do algoritmo SVM é bem intuitivo. Hui Xue, Qiang Yang e Songcan Chen, autores do capítulo 3 do livro Top 10 [27], ilustram este fato apresentando o singelo mecanismo do classificador (SVC)(*Support Vector Classifier*) usando apenas duas classes: a classe 0 e a classe 1.

O objetivo é encontrar a função de classificação que melhor divida estas duas classes. Para alcançar isso, a idéia é maximizar a margem (separação) entre as classes, o que geometricamente corresponde a um hiperplano dado por:

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{2.11}$$

Onde $\mathbf{w}$ é chamado de vetor peso, no sentido estatístico, ou seja, para dar maior ou menor peso a cada uma das suas dimensões e b é chamado de tendência (*bias*)

Podemos ainda definir a largura desta margem como a menor distância r entre os pontos mais próximos de classes distintas, expressa por:

$$r = \frac{\mathbf{w}^T\mathbf{x} + b}{||\mathbf{w}||} = 0 \tag{2.12}$$

Onde $\mathbf{w}^T\mathbf{x} + b = \mathbf{g(x)}$; é chamada de *função discriminante* ou também de *função margem* dado $\mathbf{w}$ e b. Sem perda de generalidade pode ser fixada $g(x) = 1$

Sendo os Dados de Treinamento $\mathbf{x}_i \in \Re^m, y_i \in \{-1, 1\}$ temos então :

$$\begin{aligned} \mathbf{w}^T\mathbf{x} + b \geq 1 \quad &\text{para} \quad y_i = +1 \\ \mathbf{w}^T\mathbf{x} + b \leq -1 \quad &\text{para} \quad y_i = -1 \end{aligned} \tag{2.13}$$

os pontos dos dados de treinamento $\{\mathbf{x}_i, y_i\}$ que estão mais próximos das margens são os que atendem as igualdades da equação anterior. Estes vetores são chamados de *vetores de suporte*.

Usando $||\mathbf{w}||^2$ em lugar de $||\mathbf{w}||$, para facilitar a otimização do problema, podemos considerar que a margem máxima do hiperplano pode ser calculado pelo seguinte problema de otimização

$$\begin{array}{lll} \min_{w,b} & \frac{1}{2}||w||^2 & \\ \text{sa:} & \mathbf{y}_i(\mathbf{w}^T\mathbf{x}+b) & \geq \quad 1 \\ & i = 1,2,\ldots,n & \end{array} \tag{2.14}$$

Esse problema se apresenta como um problema primal de otimização não linear restrita. No contexto da literatura de SVM, a solução deste problema normalmente é obtida pelo método dos multiplicadores de Lagrange, usando a função de Lagrange

$$L(\mathbf{w},b,\alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{n}\alpha_i[(y_i(\mathbf{w}^T\mathbf{x}+b)] \tag{2.15}$$

pseudo-código

Algoritmo 2.1.6: SVM : Suport Vector Machines

Data: Dados de Treinamento D $= \{(X,y)\}, \alpha$ parcialmente ou sem treino

Result: retorne somente os vetore de suporte $\alpha_i > 0$

initialization;

$C = C_O$ algum valor pequeno;

while $\Delta\alpha > \varepsilon$ **do**

  foreach $\{X_i,y_i\},\{X_j,y_j\}$ **do**

    optimize $\alpha_i\alpha_j$;

Vantagens e Desvantagens

Os autores do artigo Top 10 [25] afirmam que atualmente o SVM é considerado um dos algortimos de *Data Mining* que apresenta um dos métodos mais robusto e preciso, entre todos os algoritmos conhecidos. Por isso, recomendam que seja usado. Sem dúvida, a terceira posição no ranking dos mais votados é decorrente de sua boa reputação.

2.1.8 EM

Introdução

Imagine o problema de modelar a distribuição dos dados numa amostra composta por subpopulações heterogêneas. Mesmo que seja conhecida a função de distribuição de cada uma destas subpopulações, descobrir os parâmetros das distribuições de cada população não é tarefa fácil.

Na Estatística, este problema é conhecido por Modelo de Misturas Finitas (*Finite Mixture Models*) e o algoritmo que enfrenta este problema é o EM (*Expectation-Maximization*).

O EM concorreu com o livro (*Finite Mixture Models*) [22] publicado em 2000, de McLachlan, que também é o autor do capítulo 5 do livro [27] e do artigo [25] sobre o algoritmo no *TOP TEN*, ficando em quinto lugar entre os algoritmos de *Data Mining*.

Segundo McLachlan [27, 25] , o trabalho de Dempster [10] de 1977 impulsionou a aplicação do EM nos campos do aprendizado de máquina, reconhecimento de padrões e *Data Mining*.

Neste trabalho [10], Dempster usa o conceito de máxima verossimilhança (*maximum likelihood*) para obter uma estimativa de quanto um parâmetro de densidade se ajusta ao modelo da mistura finita numa dada amostra.

Mas o que é verossímil? E como isto pode nos ajudar? Verossimilhança (do latin *verisimile*) é um adjetivo que indica que algo parece ser verdade. Ou seja, a aderência de uma hipótese a realidade.

Uma Função de Máxima Verossimilhança é uma função que faz uma hipótese ter a máxima probabilidade de ser verdade. Neste trabalho, estamos interessados em encontrar parâmetros de densidade de uma mistura finita.

Respeitando a notação usada por McLachlan, seja dada uma função densidade de probabilidade f_θ governada por um conjunto de parâmetros θ. Seja $f(D;\theta)$ a probabilidade deste conjuto de parâmetros θ ocorrerem para os Dados Observados (D). Sejam as subsubpopulações $y_i \in D = \{y_1, \ldots, y_n\} \subset \Re^p$ termos independentes e identicamente distribuídos (i.i.d.).

Neste contexto, define-se a função de verossimilhança por:

$$L(\theta;D) = f(D;\theta) = \prod_{j=1}^{p} f(y_j;\theta) \tag{2.16}$$

Para resolver o problema da máxima verossimilhança (*maximum likelihood*) ou ML é necessário

calcular o parâmetro θ que maximiza a função $L(\theta;D)$ para um dado conjunto de observações D.

A solução deste problema é representada por:

$$\theta^* = \arg\max_{\theta} L(\theta;D) \tag{2.17}$$

Podemos usar esta equação para estimar os parâmetros θ^* da função de distribuição de probabilidade normal $N(\mu,\sigma)$ numa amostra de objetos de D tal que $x_i \in D = \{x_1,\ldots,x_n\} \subset \Re$.

Resolvendo esta equação chegamos às fórmulas clássicas da média (μ):

$$L(\mu;D) = \mu^* = 1/n\sum_{i=1}^{n} x_i \tag{2.18}$$

e do desvio padrão (σ) :

$$L(\sigma;D)^2 = (\sigma^*)^2 = 1/n\sum_{i=1}^{n}(x_i-\mu)^2 \tag{2.19}$$

Vejamos agora o cenário dos modelos de mistura finita. Neste caso temos mais de uma subpopulação misturada. Digamos que nossa amostra seja formada por n observações de g grupos distintos, distribuídos com uma proporção $\pi_1,\ldots,\pi_g$, tais que:

$$\sum_{1}^{g} \pi_i = 1 \tag{2.20}$$

Consideremos ainda que, $y_j \in D = \{y_1,\ldots,y_n\} \subset \Re^p$ corresponde a uma dada subpopulação j , podemos definir a função densidade da mistura y_j por:

$$f(y_j;\Psi) = \sum_{i=1}^{g} \pi_i f(y_j;\theta_i) \tag{2.21}$$

Onde a componente i da densidade $f_i(y_j;\theta_i)$ é especificada pelos parâmetros θ_i e que o vetor com todos os parâmetros desconhecidos Ψ pode ser expresso por:

$$\Psi = (\pi_1,\ldots,\pi_{g-1},\Theta_1^T,\ldots,\Theta_g^T) \tag{2.22}$$

Podemos usar uma estimativa de Ψ para agrupar os dados observados em g grupos com

probabilidade *a posteriori*, dada por:

$$\tau_i f_i(y_j;\Psi) = \frac{\pi_i f(y_j;\theta_i)}{f(y_j;\phi)} \tag{2.23}$$

O vetor de parâmetros Ψ pode ser estimado pelo ML (*Maximum Likelihood*). Este estimador, chamado MLE (*ML Estimate*) e denotado por Ψ^*, pode ser obtido extraindo as raízes da equação :

$$\frac{\partial L(\Psi)}{\partial \Psi} = 0 \tag{2.24}$$

Aqui, é conveniente substituir na equação a função de verossimilhança $L(\Psi)$ pela função log verossimilhança $\log L(\Psi)$, também conhecida por entropia cruzada (*cross-entropy*), que é uma função monotônica e apresenta vantagens analíticas e computacionais para o cálculo de máximo/mínimo dada por:

$$\log L(\Psi) = \sum_{j=1}^{n} \log f(y_j;\Psi) \tag{2.25}$$

A equação para o cálculo do estimador de máxima verossimilhança fica:

$$\frac{\partial \log L(\Psi)}{\partial \Psi} = 0 \tag{2.26}$$

que é a função log de verossimilhança (log *likelihood*), cujas soluções são os máximos locais obtidos pelo algoritmo EM.

Pseudo-Código

Algoritmo 2.1.7: EM : Expectation-Maximization para Agrupamento (*clustering*)

Data: Dados Observados D, g número de clusters , a estimativa inicial de $\Psi = \{\pi_g, \mu_{gXn}, \sum_{gXn})\}$ e ε precisão (critério de parada)

Result: g clusters

inicialização;

$k = 0$;

while $\Psi^{k+1} - \Psi^k > \varepsilon$ **do**

  E-Step;

  foreach $i \in \{1, \ldots, g\}$ **do**

    foreach $y_j \in D = \{y_1, \ldots, y_n\}$ **do**

      $\tau_{ij}^k = \frac{\pi_i^k \phi(y_j; \mu_i^k, \Sigma_i^k}{f(y_j; \Psi^k)}$;

  M-Step;

  foreach $i \in \{1, \ldots, g\}$ **do**

    $T_i^k = \sum_{j=1}^n \tau_{ij}^k$;

    $\pi_i^{k+1} = \frac{T_i^k}{n}$;

    foreach $y_j \in D = \{y_1, \ldots, y_n\}$ **do**

      $\mu_i^{k+1} = \frac{\tau_{ij}^k y_j}{T_i^k}$;

      $\sum_i^{k+1} = \frac{\tau_{ij}^k (y_j - \mu_i^{k+1})(y_j - \mu_i^{k+1})^T}{T_i^k}$

  $k = k + 1$;

MacLachlan apresenta um exemplo de aplicação em *clustering* do algoritmo EM, num cenário em que temos particular interesse: o de uma mistura finita de distribuições normais, típica de estudos com células em citometria de fluxo.

Para distribuições normais, a função log da máxima verossimilhança pode ser expressa em função dos parâmetros da densidade de uma distribuição normal $\theta = \{\mu, \sum\}$.

As premissas permanencem as mesmas, ou seja:

Temos g subpopulações misturadas numa amostra com n observações distribuídas com uma proporção $\pi_1, \ldots, \pi_g$, tais que:

$$\sum_1^g \pi_i = 1 \tag{2.27}$$

Consideremos ainda que $y_j \in D = \{y_1, \ldots, y_n\} \subset \Re^p$ corresponde a uma dada subpopulação

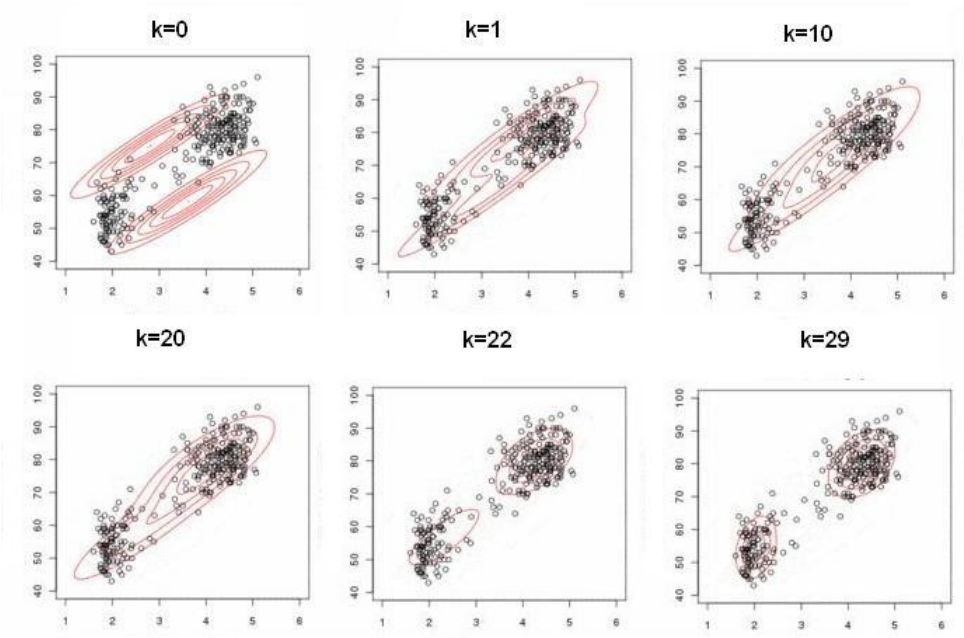

Figura 2.1: EM em Ação para k iterações

j, podemos definir a função densidade da mistura y_j por:

$$f(y_j;\Psi) = \sum_{i=1}^{g} \pi_i f(y_j;\theta_i) \tag{2.28}$$

Onde a componente i da densidade $f_i(y_j;\theta_i)$ é especificada pelos parâmetros θ_i e que o vetor, com todos os parâmetros desconhecidos Ψ, pode ser expresso por:

$$\Psi = (\pi_1,\ldots,\pi_{g-1},\Theta_1^T,\ldots,\Theta_g^T) \tag{2.29}$$

Assim, uma mistura finita com funções de distribuição normal multivariada $\phi(y_j;\mu_i,\sum_i)$, média μ_i e com a matriz de covariância $\sum_i$, tem seu estimador de Máxima Verossimilhança (MLE) dado pela função Log ML:

$$\log L(\Psi) = \sum_{j=1}^{n} \log \sum_{i=1}^{g} \pi_i \phi(y_j;\mu_i,\Sigma_i) \tag{2.30}$$

Neste cenário, consideramos a ocorrência de dados que estão faltando (*missing*) na amostra que serão indicados por z_{ij}, onde $i = 1,\ldots,g$ é o índice ligado as diversas subpopulações presentes na amostra, ou seja, os g grupos (*clusters*) e o índice $j = \{1,\ldots,n\}$, ligado a y_j, com distribuição $\phi(y_j;\mu_i,\sum_i)$.

$$\log L(\Psi) = \sum_{i=1}^{g} \sum_{j=1}^{n} z_{ij}(\log \pi_i + \log \phi(y_j; \mu_i, \Sigma_i)) \tag{2.31}$$

No passo E-step, de Expectativa (Expectation), calcula-se a probabilidade *a posteriori* de que o conjunto de parâmetros estimados se ajuste aos dados observados em cada sub-população y_j, expresso por :

$$\tau_{ij}^k = \frac{\pi_i^k \phi(y_j; \mu_i^k, \Sigma_i^k)}{f(y_j; \Psi^k)} \tag{2.32}$$

No passo M-step, de Maximização, estas estimativas são aplicadas à função log máxima verossimilhança, para calcular os estimadores que maximizam esta função. Computacionalmente é conveniente usar uma estatística suficiente $T(X)$ para cada um dos parâmetros envolvidos , $\{\pi_i, \mu_i, \sum_i\}$

$$T_{i\pi}^k = \sum_{j=1}^{n} \tau_{ij}^k \tag{2.33}$$

$$T_{i\mu}^k = \sum_{j=1}^{n} \tau_{ij}^k y_j \tag{2.34}$$

$$\pi_i^{k+1} = \frac{T_{i\pi}^k}{n} \tag{2.35}$$

$$\mu_i^{k+1} = \frac{T_{i\mu}^k}{T_{i\pi}} \tag{2.36}$$

$$\Sigma_i^{k+1} = \frac{\tau_{ij}^k (y_j - \mu_i^{k+1})(y_j - \mu_i^{k+1})}{T_i^k} \tag{2.37}$$

O algoritmo itera $k+1$ vezes nestes dois estágios que se alternam, até atingir uma condição de parada, controlada pela variável ε (*threshold*).

Vantagens e Desvantagens

McLachlan [22, 27, 25] afirma que EM é muito aplicado numa gama de problemas do tipo *dados incompletos*, pois a premissa de que os dados estão incompletos está presente na formulação original do algoritmo.

2.1.9 Ada Boost

O *Ada Boost* concorreu ao **Top Ten** com o artigo *A decision-theoretic generalization of on-line learning and an application to boosting* de Freund e Schapire [13] , no qual os autores propõem um algoritmo que fosse simples o bastante para ser implementado em ”cerca de 10 linhas de código ...”.

Segundo Zhi-Hua Zhou e Yang Yu, autores do capítulo 7 sobre o *AdaBoost* do livro **Top Ten** [27], esta simplicidade, combinada à sólida fundamentação teórica e grande precisão na predição, faz do algoritmo um dos mais importantes dentre os algoritmos do Aprendizado de Agrupamento de Máquinas (*Ensamble Learning*.

O campo de Aprendizado de Agrupamento (ou Comitê) de Máquinas (*Ensamble Learning*) pesquisa métodos que empregam múltiplos aprendizes (algoritmos de aprendizado de máquina) para resolver um problema. A capacidade de generalizar de um grupo de maquinas é muito superior ao de uma única máquina. Isto torna estes métodos muito atrativos.

A pesquisa que conduziu ao desenvolvimento do *Ada Boost*, foi motivada pela busca da resposta a seguinte pergunta: Pode um conjunto de algoritmos baseado em aprendizado fraco (*weak learner*) ter o desempenho impulsionado (*boosting*) de modo atingir o desempenho de um aprendizado forte (*strong learner*) ?

Esta questão, foi introduzido por Michael Kearns (1988) num trabalho não publicado *Thoughts on Hypothesis Boosting*, usa o conceito de hipótese de *boosting*. O algoritmo opera, a grosso modo, pela combinação dos acertos de cada um dos classificadores do conjunto de máquinas, proporcionando um ajuste (*adaptive*) no vetor de peso α_t que melhora a performance de acertos do conjunto de máquinas a cada rodada de treinamento.

pseudo-código

Ada(ptive) Boosting, é um meta-algoritmo que promove a precisão de um algoritmo baseado em aprendizado fraco *Weak Learner* tanto quanto se queira. Na extensão do algoritmo apresentada no capitulo 7 do livro [27] de onde obtemos o pseudo-código temos como entrada um algoritmo de aprendizado fraco *Weak Learner* como k-NN ou Navie Bayes.

O algoritmo usa uma base de treinamento D, composta por m pares de objetos $(x_i, y_i) \in \{(x_1, y_1), \ldots, (x_m, y_m)\}, x_i \in X \subset \Re^n, y_i \in Y = \{-1, +1\}$, onde x_i é um objeto com n atributos e y_i é o *label* que indica a qual das duas classes o objeto pertence .

Note que, a classe ser positiva ou negativa, é uma maneira simples de obter a resposta para

a Hipótese $y = H(x)$, pois basta apurar o sinal de $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$, para saber se o objeto pertence a classe positiva ou negativa.

O algoritmo procura então descobrir durante T rodadas de treinamento qual é o peso adequado α_t a ser considerado entre as rodadas de treinamento que melhor ajusta (adapta) ao erro ε_t.

Este erro representa a probabilidade $Pr_{x\sim D_{t,y}} I[h_t(x) \neq y]$ da classificação obtida para um objeto x_i estar errada $h_t(x) \neq y$ dentro do conjunto de treinamento $D_{t,y}$ utilizado na rodada t.

O algoritmo implementa uma estratégia adaptativa a cada ciclo, promovendo um ajuste no conjunto de treinamento $D_{t+1}(i)$ para rodada seguinte $t+1$

Algoritmo 2.1.8: AdaBoost

Data: um algoritmo de aprendizado L, Base de Dados $D = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
,Numero de ciclos de treinamento T

Result: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$

initialization;

$D_1(i) = 1/m$;

for $t = 1, \ldots, T$ **do**

$h_t = L(D, D_t)$% Treina a base D com um algoritmo fraco L ;

$\varepsilon_t = Pr_{x\sim D_{t,y}} I[h_t(x) \neq y]$% medida de erro;

if $\varepsilon_t > 0.5$ **then**

termina;

$\alpha_t = 0.5 ln(\frac{1-\varepsilon_t}{\varepsilon_t})$;

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} \exp(-\alpha_t y_i h(x_i)), & \text{se} \quad h_t(x_i) = y_i \\ \exp(\alpha_t y_i h(x_i)), & \text{se} \quad h_t(x_i) \neq y_i. \end{cases}$$

Vantagens , Desvantagens , principais aplicações e oportunidades

Uma das principais vantagens do método é a de aliar as vantagens presentes nos algoritmos baseados em aprendizado (*learner*) fraco (*weak*), tais como K-NN e Naive Bayes, com a performance e precisão dos algoritmos baseados em aprendizado forte (*strong*), tais como SVM e C4.5. Será o melhor de dois mundos? Parece que, em alguns campos, sim.

Este algoritmo tem sido utilizado com muito sucesso, por exemplo, no reconhecimento de faces, uma das mais promissoras aplicações de visão computacional, que é amplamente utilizada nas redes sociais e com potencial de aplicação na área de segurança.

Outro campo onde os autores do artigo indicam um potencial para trabalhos a serem desen-

volvidos é o de aplicar o *AdaBoost* na seleção de características (*features selection*).

Uma das mais claras desvantagens do método é o preço que precisa ser pago durante o treinamento já que são necessárias várias rodadas de treinamento. Outra limitação é que este algoritmo não teve na regressão o mesmo sucesso que conseguiu na classificação.

2.1.10 Apriori

O Apriori é um algoritmo originário da área de pesquisa de Banco de Dados. O trabalho com que concorreu ao *Top Ten*, foi apresentado na vigésima conferencia de VLDB em 1994, por de Agrawal e Srikant no artigo [1] *Fast algorithms for mining association rules*.

Neste artigo , os autores apresentam uma solução para o problema da lista de itens freqüentes. Este problema clássico que tem por objetivo identificar nas listas de compras de seus clientes, itens que frequentemente são comprados em conjunto.

Segundo os autores, Hiroshi Motoda e Kouzou Ohara, tanto do artigo [25] quanto do capítulo 4 do livro [27] sobre o APRIORI no *Top Ten*, a introdução desta técnica teve um impacto tão grande que a primeira coisa que um analista de Data Mining pensa em fazer para resolver um problema de itens freqüentes é aplica-la.

De fato, não se pode negar a importância que o problema de itens freqüentes tem para o mundo corporativo.

Um caso clássico citado por diversos autores, como por exemlpo, Han [16], é o da cadeia de lojas do Wallmart, onde cerveja e fraldas são colocadas em prateleiras próximas. Isto porque, verificou-se um aumento da vendas destes itens nesta configuração de loja.

Atualmente, nas principais lojas virtuais, existe grande interesse em algoritmos que permitam descobrir itens que possam interessar ao cliente pela análise do que existe no seu carrinho de compras.

pseudo-código APRIORI

Algoritmo 2.1.9: APRIORI

Data: Dados de Treinamento D

Result: $U_k F_k$

initialization;

$F_1 = frequente1 - itens$;

for $k = 2, F_{k-1} \neq 0, k++$ **do**

  $C_k = apriorigen(F_{k-1})$% gera novos candidatos;

  foreach $t \in D$ **do**

    $C_t = selecione(C_k, t)$% cadidatos contidos em t ;

    foreach *candidato* $c \in D$ **do**

      $c.count++$

    $F_k = \{c \in C_k | c.count \geq minsup\}$;

Vantagens , desvantagens , aplicações e oportunidades

Segundo os autores do artigo, a mais importante evolução do APRIORI foi apresentada em 2000 por HAN, Jiawei et all [17] com o método *FP-Growth* (*frequent pattern growth*).

Este método usa uma estratégia de dividir e conquistar para compactar os itens freqüentes numa estrutura de dados chamada de *FP-tree*, divide os dados em conjuntos associados a cada item. Isto permite o processamento dos itens em separado.

Apesar da opinião dos autores, o *FP-Growth* concorreu entre os 18 candidatos, mas não ficou entre os *Top Ten*.

2.1.11 PageRank

O algoritmo *Page Rank* concorreu ao *Top Ten* com o trabalho apresentado por Sergey Brin and Larry Page na sétima conferencia do www em 1998 : *The anatomy of a large-scale hypertextual Web Search Engine* [6].

O objetivo do algoritmo é atribuir uma importância (*Rank*) para cada página encontrada pelo mecanismo de busca na web (*Web Search Engine*).

Segundo os autores do capítulo 6 do livro [27], Bing Liu e Philip S. Yu, o sucesso do mecanismo de busca do *Google*, baseado no *Page Rank*, transformou o algoritmo no modelo dominante da área de Análise de Links.

Os autores do *Page Rank* criaram um sistema de votação para medir o prestígio (*Rank*) de uma página com as seguintes premissas:

- Uma página é votada por outra quando há um hiperlink ligando-as.
- Para o voto ser válido é necessário que as páginas pertençam a domínios (sites) diferentes (para não computar links de navegação entre páginas do mesmo domínio).
- Cada página *transfere* seus votos para as páginas apontadas. Dito de outra forma, o peso do voto de uma página depende da sua votação.

Assumindo estas hipóteses podemos desenvolver uma fundamentação matemática para o algoritmo.

Seja $G = (V,E)$ o grafo orientado da WEB onde V são os vértices e são E os arcos orientados que as ligam.

Seja $n = |v|$ o numero total de páginas.

Podemos construir um índice de pontuação (*score*) da pagina i indicado por $P(i)$ definido por :

$$P(i) = \sum_{(i,j)\in E} \frac{P(j)}{O_j} \tag{2.38}$$

Onde O_j representa o número de hiperlinks que saem da página j para domínios diferentes do domínio a que pertence.

Seja $\mathbf{A}_{ij}$ a matriz adjacente do grafo, ou seja:

$$|\mathbf{A}_{ij}| = \begin{cases} \frac{1}{O_i} & \text{se}(i,j) \in E \\ 0 & \text{Caso contrario.} \end{cases} \tag{2.39}$$

Podemos montar então um sistema de n equações, algebricamente representado por:

$$\mathbf{A}^T\mathbf{P} = \mathbf{P} \tag{2.40}$$

que é a clássica equação de autovetores $\mathbf{A}^T\mathbf{P} = \lambda\mathbf{P}$ quando $\lambda = 1$ e $\mathbf{P}$ é um auto vetor.

Esta equação se resolve pelo também clássico método da iteração de potenciação (*power iterartion*).

Para atender a requisitos de solução desta equação, introduz-se um parâmetro d e um vetor identidade $\mathbf{e}$ (todos valores são 1) na equação transformando-a em:

$$\mathbf{P} = (1-d)\mathbf{e} + d\mathbf{A}^T\mathbf{P} \tag{2.41}$$

Ou seja, chega-se a fórmula original do trabalho:

$$P(i) = (1-d) + d\sum_{(i,j)\in E} \frac{P(j)}{O_j} \tag{2.42}$$

Equivalente à fórmula:

$$P(i) = (1-d) + d\sum_{j=1}^{n} A_{ij}P(j) \tag{2.43}$$

o parâmetro d é chamado fator de *damping*

pseudo-código

Algoritmo 2.1.10: PageRank

Data: G Grafo de HiperLinks da web

Result: P(pagina) - Pontuação das Paginas

inicialização;

$\mathbf{P}_0 = \mathbf{e}/n$;

$K = 1$;

while $||\boldsymbol{P}_k - \boldsymbol{P}_{k-1} < \varepsilon||$ **do**

  $\mathbf{P}_k = (1-d)\mathbf{e} + \mathbf{A}^T\mathbf{P}_{k-1}$;

  $k = k+1$;

2.1.12 k-*Means*

O algoritmo de *Clustering*: k-*Means*, concorreu com o artigo de J. McQueen [21] publicado em 1967, ficando em segundo lugar entre os algoritmos de *Data Mining*.

No contexto de aprendizado de Máquina, algoritmos de *Clustering* resolvem problemas de Aprendizado não Supervisionado, ou seja, sem a necessidade de uma Base de Treinamento com objetos classificados.

Joydeep Ghosh e Alexander Liu, autores do tópico sobre o k-*Means* no artigo [25] e no livro [27] *Top Ten* algoritmos de *Data Mining*, resumem o algoritmo como um método para particionar uma Base de Dados num dado k número de grupos, usando para isso algum critério de proximidade entre os membros de cada grupo.

Se usarmos como critério de proximidade a métrica euclidiana, a função custo z se apresenta como:

$$z = \sum_{i=1}^{n} (\min_{j} ||x_i - c_j||_2^2) \tag{2.44}$$

onde $x_i \in D, i \in \{1,\ldots,n\}$ são as observações da base de dados D , c_j é o ponto central do j-ésimo grupo e $j \in \{1,\ldots,k\}$

Pseudo-Código

Algoritmo 2.1.11: k-means

Data: Dataset D e numero de clusters k

Result: conjunto de pontos representantes C e seus membros m

/*Inicializa todos k pontos representantes de C*/;

escolha aleatoriamente k pontos do Dataset D como representates de C ;

while $\sum_{i=1}^{n} (\min_j ||x_j - c_j||_2^2)$ *convergir* **do**

    Realoque os pontos x_j aos cluster mais próximo;

    Recalcule os centros c_j do clusters ;

O algoritmo é descrito pelos autores como uma iteração em duas etapas após a inicialização dos pontos centrais.

Na primeira etapa, cada uma das observações é alocada ao ponto central mais próximo. Na etapa seguinte, o ponto central de cada grupo é recalculado.

O algoritmo repete estas etapas até que não haja variação no mínino obtido pela função de custo.

Vantagens e Desvantagens

Os autores alertam que é uma característica do problema de *clustering* com minimização da soma dos mínimos quadrados é a ocorrência de uma quantidade expressiva muitos mínimos locais. Como consequência natural disto, o algoritmo é sempre muito sensível aos pontos iniciais.

Por esta razão, pode convergir com baixa qualidade para pontos visivelmente inadequados.

3 *Metodologia*

3.1 *Clustering*

3.1.1 Introdução

A análise de *clustering* é uma abordagem com grande aplicação em Datamining. Ao contrário dos algoritmos de aprendizado supervisionado que exigem uma base de treinamento para processar o aprendizado, os algoritmos de *Clustering* não requerem nem a base de treinamento nem uma etapa de processamento de aprendizado.

Isto se traduz na vantagem de chegarmos imediatamente a um resultado. Entretanto, por outro lado, exige-se um critério de qualidade claro para interpretação dos resultados.

Existem vários critérios de qualidade que podem ser adotados. Entre eles, um dos mais utilizados é o de homogeniedade entre os membros do cluster e de separação entre os clusters. Esta definição encontra-se em Hartingan [18].

Porém, existem outras maneiras de se expressar este critério. Han [16], por exemplo, prefere usar a similaridade ao invés da homogeniedade. Apesar de usarem definições diferentes, ambas seguem a ideia de agrupar objetos que são parecidos.

Ser parecido também pode ser expresso por diversas métricas. Na metodologia que vamos adotar , ser parecido é dado pela distância entre pontos do espaço $\Re^n$, expresso na norma euclidiana $||\bullet||_2$.

A metodologia utilizada neste trabalho, proposta por Xavier em seu artigo [28], procura minimizar a soma dos mínimos quadrados entre as observações e os centros dos grupos usando uma função de suavização.

Primeiramente vamos discutir o problema min-sum-min. Em seguida iremos introduzir a função de suavização e finalmente apresentaremos o método.

3.1.2 O Problema Min-Sum-Min

A formulação do problema de agrupamento como um problema min-sum-min é apresentada por Xavier [28] como se segue.

Seja $S = \{s_1, \ldots, s_m\}, s_i \in ||\Re^n||_2$, o conjunto das m observações representadas num espaço euclidiano com n dimensões. Estas observações devem ser agrupadas em $q \in \Re$ grupos disjuntos, em torno dos $x_i \in \Re^n, i = \{1, \ldots, q\}$ pontos centrais. Definimos também $X_k \in \Re^{nq}$ como sendo a matrix que representa as n coordenadas de cada um desses q pontos centrais na k-ézima iteração.

Para cada uma das observações s_j, podemos definir z_j como a função distância norma 2 ao centro mais próximo $x_i \in X$ pela equação :

$$z_j = \min_{x_i \in X} ||s_j - x_i||_2 \tag{3.1}$$

Adotando o somatório dos mínimos quadrados como o critério de homogeniedade entre os membros do grupo , podemos medir esta homogeniedade pela equação:

$$H(X) = \sum_{j=1}^{m} z_j^2 \tag{3.2}$$

O conjunto das possíveis localizações dos centros dos grupos X é dada por

$$\dot{X} = \arg \min_{X \in \Re^{nq}} H(X) \tag{3.3}$$

substituindo $H(X)$ nesta equação , chegamos a formulação do min-sum-min :

$$\dot{X} = \arg \min_{X \in \Re^{nq}} \sum_{j=1}^{m} \min_{x_i \in X} ||s_j - x_i||_2 \tag{3.4}$$

que pode ser resolvido pela otimização do seguinte problema:

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} z_j^2 \\ \text{sa:} \quad & z_j = \min_{i=1,2,\ldots,q} ||s_j - x_i||_2 \\ & j = 1, 2, \ldots, m \end{aligned} \tag{3.5}$$

Xavier em [28] usa uma estratégia de relaxação para atacá-lo. Assim ele reescreve a restrição seguindo os seguintes passos:

$$
\begin{aligned}
& z_j - ||s_j - x_i||_2 \leq 0 \\
& i = 1,2,\ldots,q \\
& j = 1,2,\ldots,m
\end{aligned} \tag{3.6}
$$

se introduzirmos a função $\varphi(y) = \max\{0, y\}$, o conjunto das restrições no índice i , associado aos centróides, podem ser colapsadas na seguinte forma:

$$
\begin{aligned}
& \sum_{i=1}^{q} \varphi(z_j - ||s_j - x_i||_2) = 0 \\
& j = 1,2,\ldots,m
\end{aligned} \tag{3.7}
$$

Finalmente , introduzindo-se uma pertubarção $\varepsilon \to 0$, podemos apresentar o problema na forma canônica:

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{m} z_j^2 \\
\text{sa:} \quad & \sum_{i=1}^{q} \varphi(z_j - ||s_j - x_i||_2) \geq \varepsilon \\
& j = 1,2,\ldots,m
\end{aligned} \tag{3.8}
$$

3.1.3 Suavização do Problema

O desenvolvimento apresentado na subseção anterior, originariamente apresentado por Xavier A. em seu artigo [28] sobre a suavização hiperbólica, chega a duas formulações: uma canônica e outra não canônica.

Dada uma solução viável do problema original (não canônica) consegue-se convergir para esta solução a partir da solução canônica quando $\varepsilon \to 0^+$.

Entretanto, ambas formulações são de difícil solução, pela presença da norma euclidiana nas restrições presentes em $\varphi(z_j - ||s_j - x_i||_2)$ que traz com ela problemas de convexidade e, por consequência, não atende as condições de otimalidade.

A solução proposta por Xavier A., em seu artigo [28], para contornar este problema, segue a estratégia de suavização da função φ pela função hiperbólica ϕ dada por :

$$\begin{aligned} \phi(y,\tau) &= (y+\sqrt{y^2+\tau^2})/2 \\ &\text{onde:} y \in \Re, \tau > 0 \end{aligned} \tag{3.9}$$

Esta função possui as seguintes propriedades :

(a) $\phi(y,\tau) > \varphi(y), \forall \tau > 0$, ou seja, é um limite superior de φ

(b) $\lim_{\tau \to 0} \phi(y,\tau) = \varphi(y)$, ou seja, converge para φ quando $\tau \to 0$

(c) $\phi(y,\tau)$ é uma função definida no espaço de funções convexas C^∞ para variável y

Para obter um problema diferenciável, ainda precisamos suavizar o componente presente na norma euclidiana $||s_j - x_i||_2$ e para isso introduzimos a função $\Theta(s_j, x_i, \gamma)$ dada por:

$$\begin{aligned} \Theta(s_j,x_i,\gamma) &= \sqrt{\gamma^2 + \sum_{i=1}^{n} (s_j - x_i)^2} \\ &\text{onde:} \gamma \in \Re^+, \gamma > 0 \end{aligned} \tag{3.10}$$

Esta função possui as seguintes propriedades :

(a) $\lim_{\gamma \to 0} \Theta(s_j, x_i, \gamma) = ||s_j - x_i||_2$, ou seja, converge para $||\bullet||_2$ quando $\gamma \to 0$

(b) $\Theta \in C^\infty$, ou seja, é convexa

substituindo estas funções no problema canônico obtem-se :

$$\begin{aligned} \min \quad & \sum_{j=1}^{m} z_j^2 \\ \text{sa:} \quad & \sum_{i=1}^{q} \phi(z_j - \Theta(s_j, x_i, \gamma)) \geq \varepsilon \\ & j = 1, 2, \ldots, m \end{aligned} \tag{3.11}$$

3.1.4 Método da Suavização Hiperbólica

Com as últimas transformações realizadas no problema min-sum-mim estamos prontos para apresentar o método de Suavização Hiperbólica, proposto por Xavier A., em seu artigo [28].

A proposta pode ser resumida por calcular :

$$\min f(x) \quad = \quad \sum_{j=1}^{m} z_j(x)^2 \tag{3.12}$$

obtido para cada $z_j(x)$ dado pelas raizes da equação :

$$\begin{aligned} h_j(z_j,x) \quad &= \quad \sum_{j=1}^{q} \phi(z_j - \Theta(s_j,x_i,\gamma)) - \varepsilon = 0 \\ & \qquad j = 1,2,\ldots,m \end{aligned} \tag{3.13}$$

3.1.5 Algoritmo Simplificado

Algoritmo 3.1.1: HSCM (Hiperbolic Smoothing Clustering Method) - Simplificado

Data: conjunto de observações S, q, valores iniciais de $x^0, \gamma^1, \tau^1, \varepsilon^1$
Result: $X \in \Re^{nXq}$ e o grupo de cada observação em S
initialization;
escolhe valores para $\rho_\gamma, \rho_\tau, \rho_\varepsilon \in (0,1)$;
while *critério de parada* **do**
  resolva : $\min f(x) = \sum_{j=1}^{m} z_j(x)^2$;
    Determinando cada z_j ;
    pela raiz da equação : $\sum_{i=1}^{q} \phi(z_j - \Theta(s_j,x_i,\gamma)) - \varepsilon = 0$;
    usando $\gamma^k, \tau^k, \varepsilon^k$;
  $\gamma^{k+1} = \rho_\gamma \gamma^k$;
  $\tau^{k+1} = \rho_\tau \tau^k$;
  $\varepsilon^{k+1} = \rho_\varepsilon \varepsilon^k$;
  $k = k + 1$;

4 Resultados Computacionais

Neste capítulo serão apresentados os resultados computacionais obtidos pela proposta metodológica apresentada no capítulo anterior.

4.1 Descrição dos Experimentos

Nesta primeira parte, alguns detalhes dos experimentos serão abordados: a configuração do computador, o compilador utilizado e a bibloteca de otimização.

4.1.1 Hardware e softwares utilizados

Para os experimentos computacionais foi utilizado um LAPTOP POSITIVO PREMIUM, com processador da Intel, Pentium Dual-Core T4300, 2.1 GHz e 4 GBytes de memória RAM. O algoritmo foi implementado na linguagem Fortran 77, utilizando-se o compilador Compaq Visual Fortran versão 6.1.0.

4.1.2 Bibloteca de otimização

Para a realização dos procedimentos de minimização irrestrita multidimensional foi utilizado o método Quase-Newton, com atualização da matriz hessiana dada pela forma BFGS, na implementação VA13C da Harwell Library, biblioteca disponibilizada gratuitamente em: http://www.cse.scitech.ac.uk/nag/hsl/

4.2 Geração de Dados Sintéticos

4.2.1 Testes com Dados Sintéticos

Uma das grandes dificuldades em *Data Mining* é avaliar o erro quando não se tem uma certeza sobre o resultado final. Para contornar esta dificuldade, vamos trabalhar com dados sintéticos gerados por um procedimento algoritmo bem definido, com parâmetros escolhidos *à priori*.

Seguindo esta estratégia, vamos gerar massas de teste usando uma função de distribuição paramétrica conhecida , processar os dados gerados com o algoritmo estudado e comparar os resultados obtidos com os resultados esperados.

Serão gerados problemas teste utilizando o modelo de mistura de densidades finita com a função Normal multivariada $\eta(\mu_i, \sum_i)$ e seus parâmetros , média μ_i e a matriz de covariância $\sum_i$ definidos *à priori*.

Neste Modelo, cada teste deverá simular um conjunto de m observações representadas num espaço euclidiano com n dimensões. O conjuntos das q médias μ_i com suas n coordenadas serão representados pela matriz $\dot{X_\mu} \in \Re^{nq}$. As observações serão distribuídas segundo a função normal multivariada em torno dessas médias utilizando uma matriz de covariância $\sum_i$ definida *à priori*

Neste caso, temos mais de uma população misturada. Digamos que a amostra seja gerada distribuindo as m observações pelas q distribuições, com uma proporção $\pi_1, \ldots, \pi_q$, tais que $\sum_1^q \pi_i = 1$.

4.2.2 Planejamento dos Testes

O planejamento dos problemas teste foi elaborado para gerar massas de testes que atendam a cenários de grupos com margem de separação. Esta decisão permite a formação de distribuições disjuntas, o que facilita a interpretação dos resultados.

Foram gerados nove problemas teste, todos com $m = 1.000.000$ de observações distribuídas em respectivamente $q = \{2,3,\ldots,10\}$ distribuições sintéticas conforme definido na seção 4;2;1. Assim o problema teste "t"foi gerado com $q = t+1$ distribuições sintéticas. Para cada teste a população m_i de cada distribuição sintética i, $i = 1,\ldots,q$ é dada pela equação:

$$m_i = \frac{1/i}{\sum_{i=1}^{q} 1/i} m \tag{4.1}$$

Aplicando esta equação montamos a seguinte tabela que apresenta a frequência de cada grupo na geração sintética dos problemas teste T01 a T09, cada um deles com o número fixo de observações de um milhão (10^6) de observações.

i	T01	T02	T03	T04	T05	T06	T07	T08	T09
1	333333	166667	100000	66667	47619	35714	27778	22222	18182
2	666667	333333	200000	133333	95238	71429	55556	44444	36364
3		500000	300000	200000	142857	107143	83333	66667	54545
4			400000	266667	190476	142857	111111	88889	72727
5				333333	238095	178571	138889	111111	90909
6					285714	214286	166667	133333	109091
7						250000	194444	155556	127273
8							222222	177778	145455
9								200000	163636
10									181818
T	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000	1000000

Tabela 4.1: Distribuição do número de observações entre os grupos dos testes T01 a T09

Método para Geração de Grupos com margem de separação

Para garantir a geração de Testes sintéticos com uma margem segura de separação entre as distribuições, adotamos o seguinte procedimento para cada um dos Problemas teste $t = 1,\ldots,9$:

- Distribuímos aleatóriamente as médias no plano. Cada componente μ_1, μ_2 se situa no intervalo $[0,1]$;
- Calculamos a Matriz de distâncias formada pelas distâncias entre o posicionamento das médias ;

- Com o objetivo de gerar distribuições suficientemente separadas para facilitar a interpretação dos resultados, atribuimos o valor do desvio padrão de cada distribuição sintética com o equivalente a um terço ($1/3$) da metade ($1/2$) da menor distância entre o posicionamento da média do grupo sintético aos seus vizinhos;

- As observações sintéticas foram geradas no **R** usando a função *rnorm()* e visualizamos o resultado usando a função *plot()*.

- redefinimos o posicionamento das médias de algumas distribuições sintéticas e/ou ajustamos o seu desvio padrão para obter o problema teste com as características desejadas.

Devido a esta estratégia, cada um dos nove cenários de testes são independentes entre si e desta forma configuram distintos graus de dificuldade, sem correlação com o número de distribuições de cada cenário programado.

Ferramentas desenvolvidas

Foram desenvolvidos Scripts R para gerar os dados síntéticos segundo método descrito acima e uma interface para capturar e tratar os logs de saída do HSCM (Hiperbolic Smoothing Clustering Method). Esta mesma interface é responsável pela transformação dos dados para permitir a integração com o módulo de análise e interpretação dos resultados em R.

4.2.3 Arquivos Gerados

A Tabela abaixo contém os parâmetros utilizados em cada teste. Na primeira coluna (*Teste*) identificamos cada um dos testes. O número da distribuição dentro do teste é identificado na coluna seguinte. Assim, para o teste $t = 1$, denominado T01, temos duas distribuições (D1 e D2), para o teste $t = 2$, T02, temos três distribuições (D1, D2 e D3), etc.

A coluna *No.Obsv* apresenta o número de observações geradas para cada distribuição do problema teste. Finalmente as colunas μ_1 , μ_2, σ_1 e σ_2 informam respectivamente a média $\mu_i = (\mu_1, \mu_2)$ e a matriz de covariância $\sum_i = (\sigma_1, \sigma_2)$.

Em relação ao primeiro problema teste T01, cabe ressaltar que, uma das distribuições sintéticas (D1) foi gerada com uma distribuição normal multivariada, ou seja na qual $\sigma_1 \neq \sigma_2$, fato que não ocorreu em nenhuma distribuição dos demais testes sintéticos, que foram gerados com a distribuição normal monovariada ($\sigma_1 = \sigma_2$).

Teste	Dist	No.Obsv	μ_1	μ_2	σ_1	σ_2
T01	D1	333333	0.83221094	0.26646263	0.05098448	0.06373060
	D2	666667	0.35407602	0.42866435	0.09559590	0.09559590
T02	D1	166667	0.50762250	0.89891049	0.04402144	0.04402144
	D2	333333	0.83156430	0.63217563	0.04402144	0.04402144
	D3	500000	1.41901475	0.57455651	0.08710450	0.08710450
T03	D1	100000	0.71851277	0.12844178	0.01758616	0.01758616
	D2	200000	0.98257442	0.10326317	0.01758616	0.01758616
	D3	300000	0.33321435	0.39597429	0.02329220	0.02329220
	D4	400000	0.63653327	0.43012709	0.02329220	0.02329220
T04	D1	66667	0.48805936	0.26413854	0.00238892	0.00238892
	D2	133333	0.40604815	0.21094234	0.00238900	0.00238900
	D3	200000	0.55141965	0.36142836	0.00337000	0.00337000
	D4	266667	0.73512097	0.37110793	0.00846000	0.00846000
	D5	333333	0.57249409	0.99113278	0.11858178	0.11858178
T05	D1	47619	0.80962436	1.12900273	0.00748817	0.00748817
	D2	95238	0.66751356	1.03022411	0.00748817	0.00748817
	D3	142857	0.40565136	0.95799192	0.01228095	0.01228095
	D4	190476	0.18992253	1.00883349	0.01228095	0.01228095
	D5	238095	0.90501802	0.61297415	0.05762648	0.05762648
	D6	285714	0.37248857	0.39504859	0.07950124	0.07950124
T06	D1	35714	0.36748482	0.92190516	0.00786188	0.00786188

	D2	71429	0.46182854	1.04747721	0.00786188	0.00786188
	D3	107143	0.46357837	0.75141464	0.00957525	0.00957525
	D4	142857	0.14068468	0.81770756	0.01557386	0.01557386
	D5	178571	0.17323695	1.29927733	0.03667210	0.03667210
	D6	214286	0.66879473	1.45756438	0.05275162	0.05275162
	7	250000	1.02556901	0.55836454	0.08827545	0.08827545
T07	D1	27778	0.22554308	0.95594568	0.00248124	0.00248124
	D2	55556	0.17146881	0.87227410	0.00248124	0.00248100
	D3	83333	0.36249773	0.38050822	0.00427000	0.00427000
	D4	111111	0.48804830	0.41677916	0.01557000	0.01557000
	D5	138889	0.68917633	0.26903284	0.01557000	0.01557000
	D6	166667	0.38800303	0.71930790	0.01757143	0.01757100
	D7	194444	0.81668704	0.98742136	0.02216080	0.02216080
	D8	222222	0.80441795	0.68994422	0.04356590	0.04356590
T08	D1	22222	0.96633445	0.51938746	0.00254764	0.00254764
	D2	44444	0.94940684	0.60515686	0.00254764	0.00254764
	D3	66667	0.98844778	0.75542811	0.00602640	0.00602640
	D4	88889	0.73359481	0.94877369	0.01139550	0.01139550
	D5	111111	0.84194312	1.13273754	0.01139550	0.01139550
	D6	133333	1.17826147	1.00045423	0.02401680	0.02401680
	D7	155556	0.76854616	0.20131717	0.03507220	0.03507220
	D8	177778	0.46738002	0.48119427	0.04225810	0.04225810
	D9	200000	0.30960653	1.05078615	0.04754320	0.04754320
T09	D1	18182	0.93221094	0.36646263	0.00047286	0.00047286
	D2	36364	0.96678479	0.39284604	0.00047286	0.00047286
	D3	54545	0.37595259	0.95838749	0.00087628	0.00087628
	D4	72727	0.33290692	0.91774033	0.00087628	0.00087628
	D5	90909	0.36149275	0.44263401	0.00399322	0.00399322
	D6	109091	0.45407602	0.52866435	0.00399322	0.00399322
	D7	127273	0.84090473	0.54439666	0.00970335	0.00970335
	D8	145455	0.20926368	0.60482277	0.01237000	0.01236972
	D9	163636	0.17537424	0.17194686	0.02697791	0.02697791
	D10	181818	1.09136311	0.90336388	0.04789672	0.04789672

Tabela 4.2: Parâmetros de Geração dos Testes com 1 milhão de observações

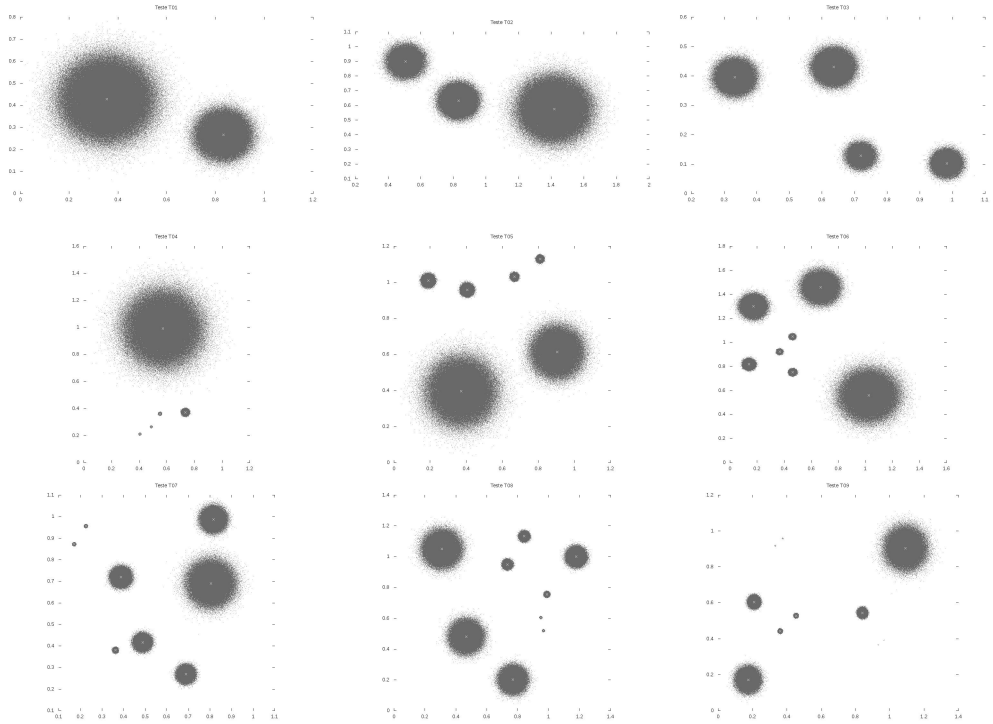

Figura 4.1: Testes Gerados de 1 a 9 com 1 milhão de observações

Os nove gráficos apresentados na figura 4.1 apresentam as observações sintéticas geradas para cada um dos 9 problemas teste gerados.

4.3 Resultados Obtidos

Nesta seção serão apresentados e analisados os resultados obtidos pelo HSCM com o processamento dos dados Sintéticos.

4.3.1 Apresentação dos Resultados dos Testes T01 a T09

Teste	q	H_{sint}	H_{best}	% $d_{sin-best}$	ocorr	D_{mean}	T_{medio}
T01	2	9111.56	9105.12	0.07%	10	0.00	1.41
T02	3	9515.86	9510.99	0.05%	10	0.00	1.81
T03	4	944.05	944.05	0.00%	10	0.00	2.18
T04	5	9415.00	6785.75	27.93%	4	0.12	4.20
T05	6	5302.48	5101.52	3.79%	4	0.01	4.46
T06	7	5664.87	5009.09	11.58%	3	0.02	4.52
T07	8	1263.91	1178.42	6.76%	1	0.02	4.35
T08	9	2131.83	1956.88	8.21%	2	0.02	4.63
T09	10	1148.11	831.85	27.55%	4	0.00	4.92

Tabela 4.3: Resultado dos dados sintéticos T01 a T09

A tabela 4.3 apresenta um resumo dos resultados obtidos utilizando-se o HSCM nas observações dos testes T01 a T09, cada um com um milhão (10^6) de pontos no $\Re^2$.

Nesta tabela tem-se a identificação do problema teste na coluna *Teste* e na coluna q , o respectivo número de grupos nos quais um milhão de observações foram particionadas.

O valor da métrica de homogeniedade, que é basicamente o valor da função objetivo (3.2), é apresentado na coluna H_{sint} para os agrupamentos sintetizados e na coluna H_{best} para os melhores resultados do HSCM.

Cabe lembrar aqui que a métrica de homogeniedade em um conjunto de pontos $X = \{x_1, \ldots, x_q\}$ adotada na metodologia é a soma dos quadrados das distâncias de cada observação s_j ao centro mais próximo de $x_i \in X$, expressa por $H(X)$ dada pela seguinte equação :

$$H(X) = \sum_{j=1}^{m} z_j^2 \; ; \tag{4.2}$$

onde z_j é a função menor distância norma 2 de cada observação s_j do teste ao centróide mais próximo dado pela equação :

$$z_j = \min_{x_i \in X} ||s_j - x_i||_2 \tag{4.3}$$

No cálculo da homogeniedade expressa por $H_{best}(X)$, foi utilizado o conjunto X, composto pelos centróides dos melhores resultados obtidos pelo algoritmo HSCM.

Por outro lado, no cálculo da homogeniedade sintética $H_{sint}(X)$, o conjunto X é composto pelos centros ($x_i = \mu_i$) dos agrupamentos sintetizados para o problema teste e neste caso z_j é dado pela equação :

$$z_j = ||s_j - \mu_i||_2 \tag{4.4}$$

A coluna $\%\ d_{sin-best}$ compara o resultado téorico ($H_{sint}(X)$) com o obtido pelo ($H_{best}(X)$) HSCM , expresso percentuamente por:

$$\%d_{sin-best} \quad = \quad 100(\frac{H_{sint} - H_{best}}{H_{sint}}) \tag{4.5}$$

Analisando seus resultados, pode-se constatar que em todos os problemas testes $t = 1, \ldots, 9$ o valor $H_{best}(X)$ foi inferior ou igual(soménte para $(t = 3)$), ao ($H_{sint}(X)$) Devemos destacar que os ganhos proporcionados pelo algoritmo HSCM para os casos $t = 1, t = 9$ foram expressivos, maiores que $27.\%$.

Conforme foi apresentado no capitulo 3, o algoritmo encontra estes pontos resolvendo um problema de otimização global. Lembrando que um problema de otimização global, em geral, apresenta uma quantidade muito grande de mínimos locais. Encontra-se na literatura, como por exemplo Bagirov [2], há a indicação para o uso de múltiplos pontos de inicialização (*multi start*) para atenuar essa dificuldade.

A versão do HSCM utilizada neste trabalho, H.26, implementa esta estratégia. Assim, para cada problema teste, foram processados dez conjuntos de pontos iniciais (*starting points*) diferentes. Os melhores resultados com dois dígitos decimais estão apresentados na tabela 4.3 na coluna (H_{best}) .

A consistência dos resultados obtidos pelo algoritmo pode ser medida pelo número de ocorrências (*ocorr*) do melhor resultado H_{best} entre os dez (*starting points*)

Outra medida de consistência é dada pela coluna do desvio médio (D_{mean}) observada nos resultados do valor de H_{best} obtidos a partir dos (*starting points*)

Por exemplo, o teste T01 atingiu sempre o melhor resultado em todos os pontos inicias, atingindo portanto um número de ocorrências igual a dez. Além disso, o desvio médio entre os

resultados de cada ponto foi de apenas $9.98E-08$, ou seja, 0.0000000998.

Assim, para o caso T01, foi observada a consistência absoluta do HSCM nas dez tentativas. De forma geral , a coluna D_{mean} mostra valores muito baios para os desvio médios, corroborando com a consistência do HSCM.

Analisando a coluna *Ocorr*, podemos observar que nos tres primeiros casos, houve $100\,\%$. Nos demais casos, essa frequência diminuiu, mas deve-se ressaltar que essa queda é decorrente da exigência de um alto nível de precisão, em 6 casas decimais. Se considerarmos 5 casas decimais, teriamos $Occur = 10$.

A coluna T_{medio} apresenta o tempo médio das dez tentativas, medidos em segundos. Os valores tabulados comprovam a eficiência do algoritmo HSCM para resolução do problema de *clustering* com $1.000.000$ (um milhão) de observações.

4.3.2 Análise dos Resultados

Resultados Problemas Testes T01,T02 e T03

A tabela 4.4 apresenta os resultados dos problemas teste T01, T02 e T03, cujo resultado obtido comportou-se dentro do esperado.

As distribuições sintéticas associadas a cada teste estão identificadas na coluna *Dist*, a posição de suas médias nas colunas (μ_1, μ_2) e o número de observações dessas distribuições na coluna (obs_{sin}) .

As soluções ou *clusters* obtidos pelo HSCM são indentificados pela coluna (Grp) , as posições de cada um dos seus centróides nas colunas (c_1, c_2) e o número de observações associadas a cada *cluster* se encontra na coluna (obs_{best}).

A ultima coluna *delta obs* exibe exibe a diferença entre o número de observações de cada distribuição sintética e o número de observações do seu *cluster* correspondente ou seja :

$$delta\ obs = obs_{sin} - obs_{best} \tag{4.6}$$

Para o teste T01 , encontram-se as informações de suas distribuções sintéticas $\{D1, D2\}$ e as soluções do HSCM $\{h1, h2\}$.

Para o teste T02 , encontram-se suas distribuções sintéticas $\{D1, D2, D3\}$ e as soluções do HSCM $\{h1, h2, h3\}$.

Finalmente, para o teste T03 , encontram-se suas distribuções sintéticas $\{D1, D2, D3, D4\}$ e as soluções do HSCM $\{h1, h2, h3, h4\}$.

Probl	Distribuições Sintéticas				Soluções HSCM				delta
Teste	Dist	μ_1	μ_2	obs_{sin}	Grp	c1	c2	obs_{best}	obs.
T01	D1	0.8322	0.2665	333333	h1	0.8318	0.2669	333647	-314
	D2	0.3541	0.4287	666667	h2	0.3538	0.4287	666353	314
T02	D1	0.5076	0.8989	166667	h1	0.5078	0.8992	166668	-1
	D2	0.8316	0.6322	333333	h2	0.8319	0.6321	333525	-192
	D3	1.4190	0.5746	500000	h3	1.4193	0.5742	499807	193
T03	D1	0.7185	0.1284	100000	h1	0.7186	0.1285	100000	0
	D2	0.9826	0.1033	200000	h2	0.9826	0.1032	200000	0
	D3	0.3332	0.3960	300000	h3	0.3332	0.3960	300000	0
	D4	0.6365	0.4301	400000	h4	0.6365	0.4300	400000	0

Tabela 4.4: Resultados dos Testes T01, T02 e T03

As representações gráficas dos resultados para estes três testes estão registrados nas figuras 4.2,4.3 e 4.4.

Cada figura mostra a representação gráfica da solução do respectivo problema teste. As média de cada distribuição normal estão assinaladas na figura pelo simbolo "+". Os parâmetros (μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As três figuras mostram os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos q igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "$\times$".

Quando a média e o centróide coincidem ocorre a fusão do símbolo "+"com o símbolo "$\times$"transformando-se em "*"

Resultado T01

Para o primeiro problema teste T01, pode-se observar que houve uma correspondencia entre cada distribuição sintética e os *clusters* da solução do HSCM. Desta forma, tem-se que a distribuição sintética D1 pode ser associado ao *cluster* h1. Analogamente , a distribuição sintética D2 associa-se ao *cluster* h2.

Pode-se observar também que as posições dos centróides obtidas pela Solução do HSCM são muito próximos das médias das distribuições sinéticas.

Analisando-se a ultima coluna *delta obs*, que exibe a diferença entre o número de observações de cada distribuição sintética e seu correspondente *cluster*, constata-se que houve uma diferença de 314 entre as Distribuições Sintéticas e as Soluções do HSCM.

Deve-se ressaltar que este resultado não se configura em um erro, pois há pontos da distribuição sintética que estão mais próximos do centro do outro *cluster*.

A alocação destas observações ao cluster mais próximo responde pelo ganho de 0.07% apresentado pelo Teste na tabela 4.3

Resultado T02

Para o problema Teste T02 foram obtidos resultados similares ao Teste T01

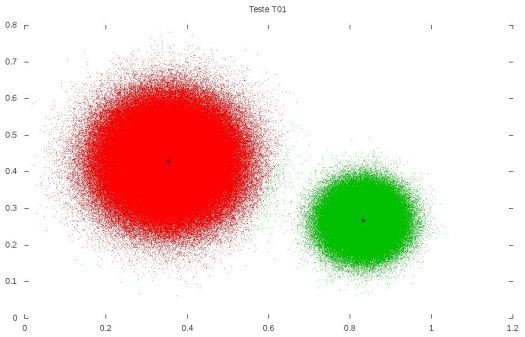

Figura 4.2: Resultados HSCM T01 - 2 *clusters*

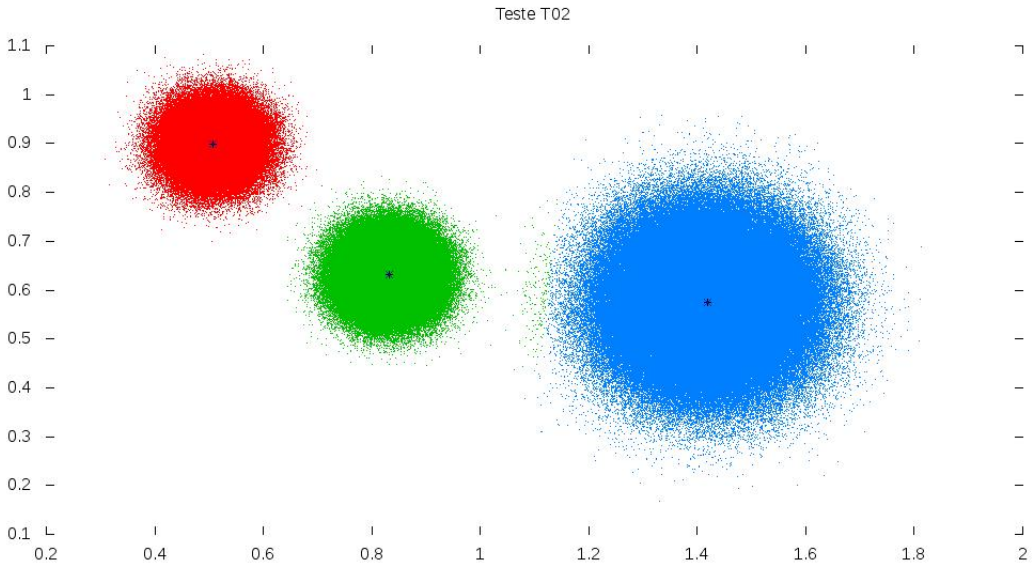

Figura 4.3: Resultados HSCM T02 - 3 *clusters*

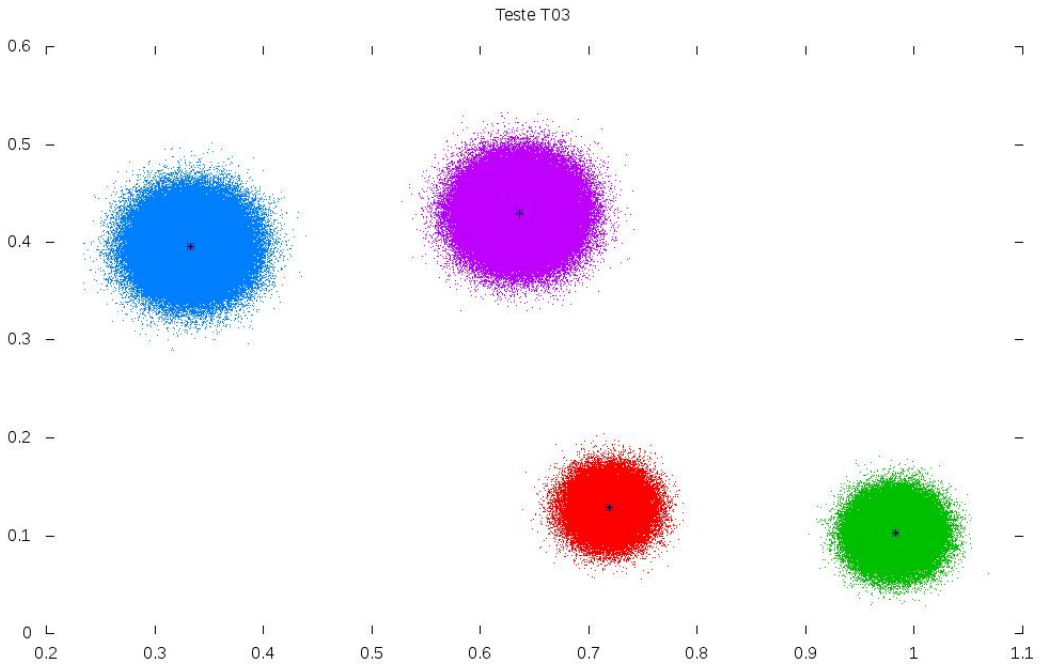

Figura 4.4: Resultados HSCM T03 - 4 *clusters*

Resultado T03

Para o Problema Teste T03, foram obtidos resultados similares aos Testes T01 e T02. A coluna delta obs mostra que neste todas as observações geradas sintéticamente foram alocadas ao *cluster* correspondente.

Resultado Teste T04

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	obs
D1	0.4881	0.2641	66667	h1	0.5513	0.3624	200717	
D2	0.4060	0.2109	133333					
			200000				200717	-717
D3	0.551400	0.361400	200000	h2	0.433400	0.228700	199999	1
D4	0.735100	0.371100	266667	h3	0.735000	0.371800	267279	-612
D5	0.572500	0.991100	333333	h4	0.478400	0.975700	165502	
				h5	0.665000	1.009700	166501	
			333333				332003	1330

Tabela 4.5: Resutados do Teste T04

A tabela 4.5 apresenta as informações sobre as 5 distribuições sintéticas (*Dist*) identificadas por $\{D1, D2, \ldots, D5\}$: a posição de suas médias (μ_1, μ_2) e o número de observações dessas distribuições (*obs*).

Esta tabela apresenta ainda informações sobr os cinco *clusters* obtidos pelo HSCM (*Grp*) indentificados por $\{h1, h2, \ldots, h5\}$: a posição dos seus centróides (c_1, c_2) e o número de observações desses grupos (*obs*).

Pode-se observar que a solução encontrada pelo HSCM, agrupou as distribuições sintéticas *D1* e *D2* em um único *cluster h1* e simultanemente particionou as observações da distribuição sintética *D5* em dois *clusters* diferentes : *h4* e *h5*.

Este efeito deve-se ao fato de que a distribuição *D5* possui uma grande variância. Os pontos mais afastados da média contribuem mais fortemente para a soma dos quadradados das distâncias. Assim, ao se particionar a distribuição sintética em dois *clusters* menores, diminui-se a variância dos *clusters* partionados e consequentemente minimiza-se a função objetivo.

Em um efeito simultâneo, a solução encontrada pelo HSCM provocou o agrupamento de duas distribuições sintéticas em um único *cluster*. Evidentemente isso produz um aumento da contribuição das observações asssociadas a estas das distribuições sintéticas ao valor da função objetivo.

Esse efeito foi amplamente compensado pela ganho decorrente do particionamento das observações da distribuição sintética D5, fato compravado pelo decrécimo de 27.93% no valor de H_{best} em relação a H_{sint} apresentado na tabela 4.3

A figura 4.5 mostra a representação gráfica da solução do respectivo problema teste T04. As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros

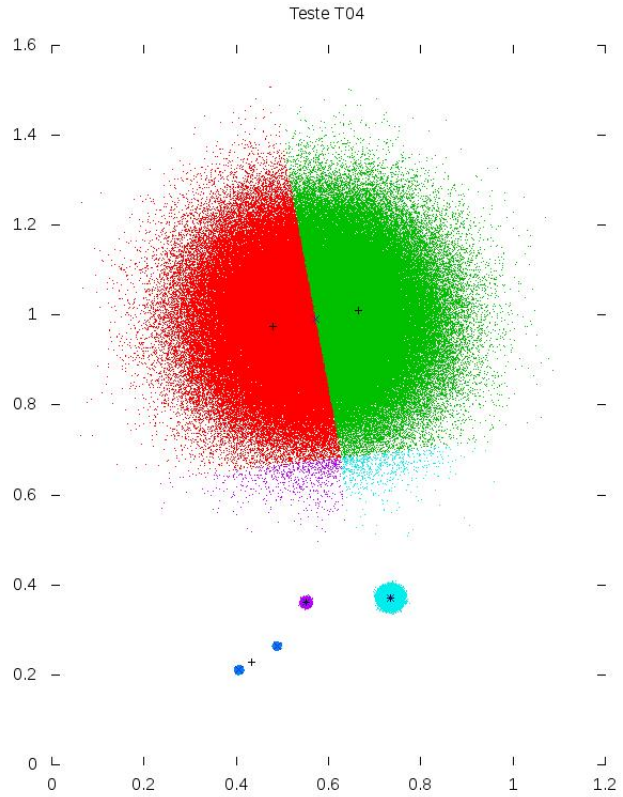

Figura 4.5: Resultados HSCM T04 - 5 *clusters*

(μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 5$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "×".

Quando a média e o centróide coincidem ocorre a fusão do símbolo "+"com o símbolo "×"transformando-se em "*"

Resultado Teste T05

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	
D1	0.8096	1.1290	47619	h1	0.7149	1.0631	142863	
D2	0.6675	1.0302	95238					
			142857				142863	-6
D3	0.4057	0.9580	142857	h2	0.4056	0.9580	142872	-15
D4	0.1899	1.0088	190476	h3	0.1899	1.0089	190476	0
D5	0.9050	0.6130	238095	h4	0.9049	0.6128	238130	-35
D6	0.3725	0.3950	285714	h5	0.3994	0.3374	142508	
				h6	0.3457	0.4524	143150	
			285714				285658	56

Tabela 4.6: Resultados do Teste T05

A tabela 4.6 apresenta as 6 distribuições identificadas por $\{D1, D2, \ldots, D6\}$ (*Dist*), o número de observações dessas distribuições (*obs*) e a posição de suas médias (μ_1, μ_2).

Apresenta ainda os 6 grupos obtidos pelo HSCM (*Grp*) indentificados por $\{h1, h2, \ldots, h6\}$, o número de observações desses grupos (*obs*) e a posição dos seus centróides (c_1, c_2)

Pode-se notar que, assim como no teste T04, a solução encontrada pelo HSCM particonou a distribuição número *D6* nos *clusters h5* e *h6* e agrupou as distribuições *D1* e *D2* no *cluster h1*.

Concorreram para isto os mesmos motivos discutidos no T04: A grande variância da distribuição sintética *D6* e a proximidade das distribuições sintética *D1* e *D2*.

A combinação deste efeitos no Teste 05 também foi favorável à solução obtida, 3.79% melhor do que o sintético. Porém este resultado foi menos expressivo que os quase 30% obtidos pelo teste T04.

A figura 4.6 mostra a representação gráfica da solução do problema teste T05 As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros (μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 6$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "$\times$".

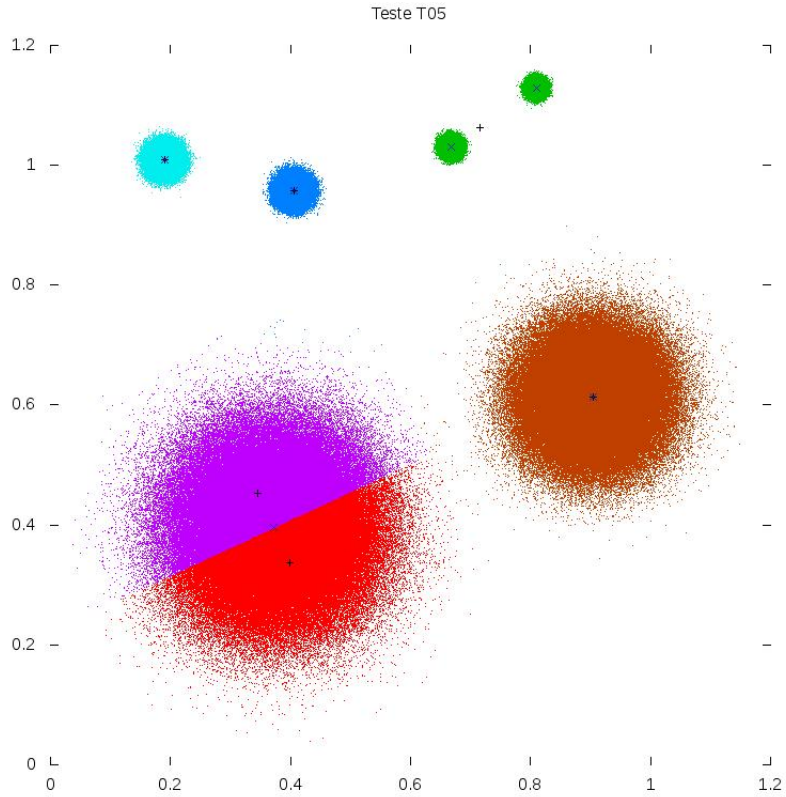

Figura 4.6: Resultados HSCM T05 - 6 *clusters*

Quando a média e o centróide coincidem ocorre a fusão do símbolo ”+”com o símbolo ”×”transformando-se em ”*”

Resultado Teste T06

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	
D1	0.3675	0.9219	35714	h1	0.4303	1.0056	107143	
D2	0.4618	1.0475	71429					
			107143				107143	0
D3	0.4636	0.7514	107143	h2	0.4636	0.7514	107167	-24
D4	0.1407	0.8177	142857	h3	0.1407	0.8177	142857	0
D5	0.1732	1.2993	178571	h4	0.1733	1.2994	178571	0
D6	0.6688	1.4576	214286	h5	0.6687	1.4573	214286	
D7	1.0256	0.5584	250000	h6	1.0964	0.5532	124827	
				h7	0.9558	0.5635	125149	
			250000				249976	24

Tabela 4.7: Resultados do Teste T06

A tabela 4.7 apresenta as 7 distribuições identificadas por $\{D1, D2, \ldots, D7\}$ (*Dist*), o número de observações dessas distribuições (*obs*) e a posição de suas médias (μ_1, μ_2).

Apresenta ainda os 7 grupos (*Grp*) obtidos pelo HSCM ($\{h1, h2, \ldots, h7\}$) , o número de observações desses grupos (*obs*) e a posição dos seus centróides (c_1, c_2)

Pode-se notar que, assim como nos problemas teste T04 e T05, a solução encontrada pelo HSCM dividiu a distribuição número *D7* nos grupos *h6* e *h7* e agrupou as distribuições *D1* e *D2* no grupo *h1*.

Concorreram para isto os mesmos motivos discutidos nos testes : A grande variância da distribuição *D7* e a proximidade das distribuições sintética *D1* e *D2*.

Aqui obteve-se um resultado mais expressivo 11.58% que o obtidos pelo teste T05 , porém inferior ao obtido pelo teste T04.

A figura 4.7 mostra a representação gráfica da solução do problema teste T06 As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros (μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 7$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "$\times$".

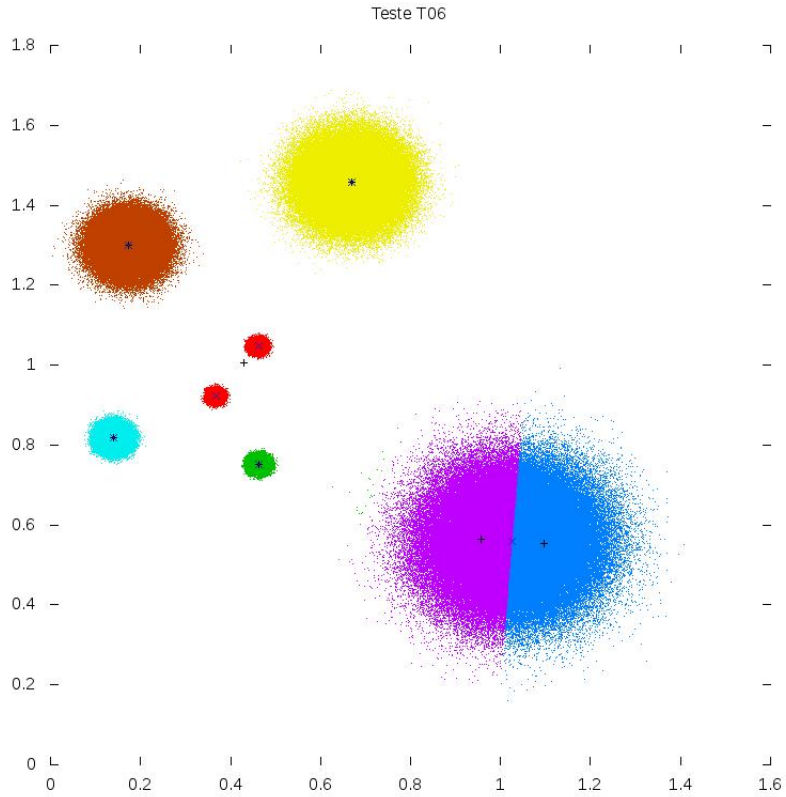

Figura 4.7: Resultados HSCM T06 - 7 *clusters*

Quando a média e o centróide coincidem ocorre a fusão do símbolo ”+”com o símbolo ”×”transformando-se em ”*”

Resultado Teste T07

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	
D1	0.2255	0.9559	27778	h1	0.1895	0.9002	83334	
D2	0.1715	0.8723	55556					
			83334				83334	0
D3	0.3625	0.3805	83333	h2	0.3625	0.3805	83334	-1
D4	0.4880	0.4168	111111	h3	0.4879	0.4169	111110	1
D5	0.6892	0.2690	138889	h4	0.6892	0.2691	138890	-1
D6	0.3880	0.7193	166667	h5	0.3880	0.7193	166667	0
D7	0.8167	0.9874	194444	h6	0.8167	0.9873	194508	-64
D8	0.8044	0.6899	222222	h7	0.7695	0.6899	110921	
				h8	0.8391	0.6902	111236	
			222222				222157	65

Tabela 4.8: Resultados do Teste T07

A tabela 4.8 apresenta as oito distribuições(*Dist*) identificadas por $\{D1, D2, \ldots, D8\}$, o número de observações dessas distribuições (*obs*) e a posição de suas médias (μ_1, μ_2).

Apresenta ainda os oito grupos $q = 8$ obtidos pelo HSCM (*Grp*) indentificados por $\{h1, h2, \ldots, h8\}$, o número de observações desses grupos (*obs*) e a posição dos seus centróides (c_1, c_2)

Concorreram para isto os mesmos motivos discutidos nos testes T04, T05 e T06 : A grande variância da distribuição *D7* e a proximidade das distribuições sintética *D1* e *D2*.

Aqui obteve-se um resultado intermediário 6.76% perto dos obtidos pelos testes T05 e T06 mas longe do expressivo quase 30% obtidos pelo teste T04.

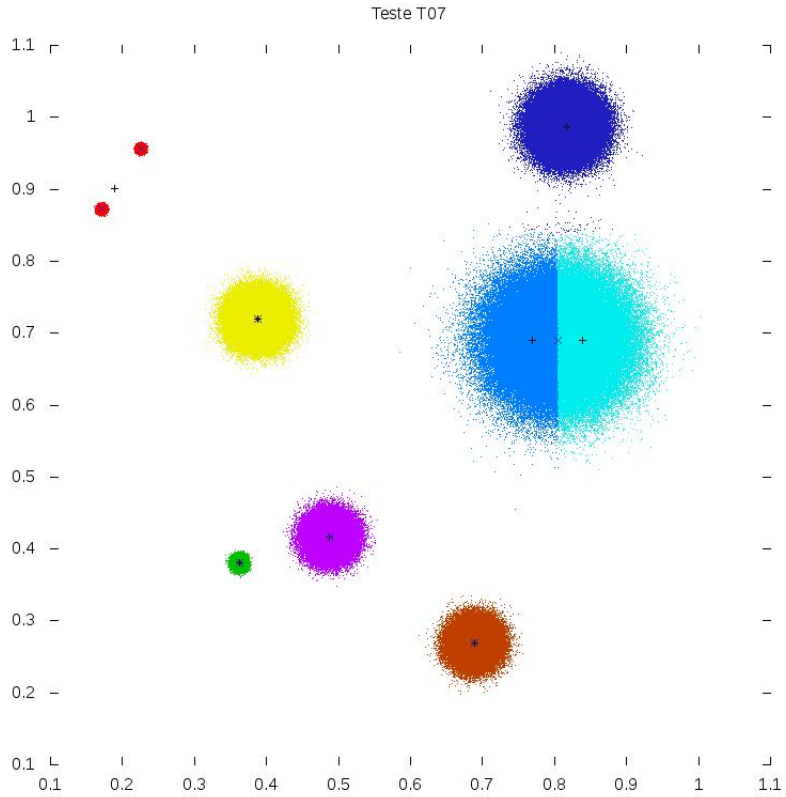

Figura 4.8: Resultados HSCM T07 - 8 *clusters*

A figura 4.8 mostra a representação gráfica da solução do problema teste T07 As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros (μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 8$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "$\times$".

Quando a média e o centróide coincidem ocorre a fusão do símbolo "+"com o símbolo "$\times$"transformando-se em "*"

Resultado Teste T08

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	
D1	0.9663	0.5194	22222	h1	0.9550	0.5766	66666	
D2	0.9494	0.6052	44444					
			66666				66666	0
D3	0.9884	0.7554	66667	h2	0.9884	0.7554	66667	0
D4	0.7336	0.9488	88889	h3	0.7336	0.9488	88889	0
D5	0.8419	1.1327	111111	h4	0.8419	1.1327	111111	0
D6	1.1783	1.0005	133333	h5	1.1783	1.0004	133333	0
D7	0.7685	0.2013	155556	h6	0.7685	0.2012	155556	0
D8	0.4674	0.4812	177778	h7	0.4674	0.4812	177778	0
D9	0.3096	1.0508	200000	h8	0.3470	1.0571	100338	
				h9	0.2721	1.0445	99662	
			200000				200000	0

Tabela 4.9: Resultados do Teste T08

A tabela 4.9 apresenta as 9 distribuições identificadas por $\{D1, D2, \ldots, D9\}$ (*Dist*), o número de observações dessas distribuições (*obs*) e a posição de suas médias (μ_1, μ_2). Apresenta ainda os nove grupos obtidos pelo HSCM (*Grp*) indentificados por $\{h1, h2, \ldots, h9\}$, o número de observações desses grupos (*obs*) e a posição dos seus centróides (c_1, c_2)

Pode-se observar que a solução encontrada pelo HSCM, dividiu a distribuição número *D9* nos *clusters h8* e *h9* e agrupou as distribuições *D1* e *D2* no *cluster H1*. Isto vem ocorrendo em todos os testes desde o T04

Este efeito deve-se ao fato de que a distribuição *D9* possui uma grande variância e a proximidade das distribuições sintética *D1* e *D2*.

Aqui obteve-se um resultado 8.21%; melhor que o resultado esperado.

A coluna *delta obs* mostra que a solução obtida pelo HSCM produziu uma perfeita colapsação das obsservações associadas às distribuições D1 e D2, assim como um perfeito particionamento da distribuição sintética D9 nos dois *clusters* h8 e h9

A figura 4.9 mostra a representação gráfica da solução do problema teste T08 As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros (μ_i, σ_i) destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

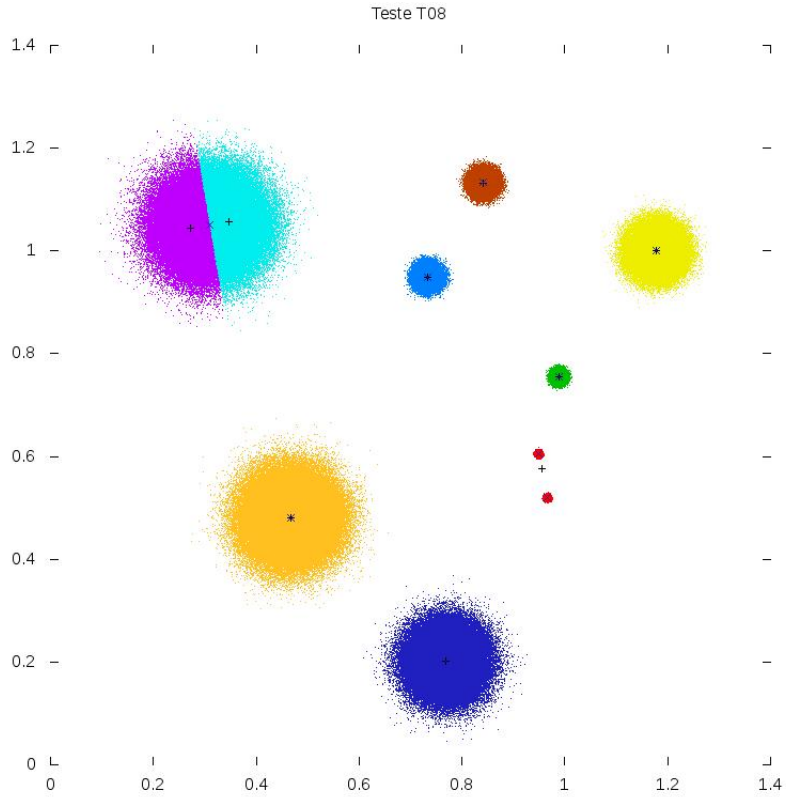

Figura 4.9: Resultados HSCM T08 - 9 *clusters*

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 9$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo ”×”.

Quando a média e o centróide coincidem ocorre a fusão do símbolo ”+”com o símbolo ”×”transformando-se em ”*”

Resultado Teste T09

Distribuições Sintéticas				Soluções HSCM				delta
Dist	μ_1	μ_2	obs_{sin}	Grp	c_1	c_2	obs_{best}	
D1	0.9322	0.3665	18182	h1	0.9553	0.3841	54546	
D2	0.9668	0.3928	36364					
			54546				54546	0
D3	0.3760	0.9584	54545	h2	0.3514	0.9352	127272	
D4	0.3329	0.9177	72727					
			127272				127272	0
D5	0.3615	0.4426	90909	h3	0.3615	0.4426	90909	0
D6	0.4541	0.5287	109091	h4	0.4541	0.5287	109091	0
D7	0.8409	0.5444	127273	h5	0.8409	0.5444	127274	-1
D8	0.2093	0.6048	145455	h6	0.2093	0.6048	145455	0
D9	0.1754	0.1719	163636	h7	0.1752	0.1720	163636	0
D10	1.0914	0.9034	181818	h8	1.0447	0.8869	60463	
				h9	1.1292	0.8708	60655	
				h10	1.1007	0.9521	60699	
			181818				181817	1

Tabela 4.10: Resultados do Teste T09

O resultado mais interessante foi obtido no Problema teste T09.

A tabela 4.10 apresenta as informações sobre as 10 distribuições sintéticas (*Dist*) identificadas por $\{D1, D2, \ldots, D10\}$: a posição de suas médias (μ_1, μ_2) e o número de observações dessas distribuições (*obs*).

Esta tabela apresenta ainda informações sobre os dez *clusters* obtidos pelo HSCM (*Grp*) indentificados por $\{h1, h2, \ldots, h10\}$: a posição dos seus centróides (c_1, c_2) e o número de observações desses grupos (*obs*).

Pode-se observar que a solução encontrada pelo HSCM desta vez agrupou duas distribuições sintéticas em um único *cluster*, *D1* e *D2* em e *h1* e *D3* e *D4* em *h2*

Simultanemente particionou as observações da distribuição sintética *D10* em três *clusters* diferentes : *h8*,*h9* e *h10*.Assim como nos testes T04 a T08, este efeito também deve-se ao fato de uma distribuição , no caso a *D10*, possuir uma grande variância.

Como os pontos mais afastados da média contribuem mais fortemente para a soma dos quadradados das distâncias. Desta forma ao se particionar a distribuição sintética em três *clusters* menores, diminui-se a variância dos *clusters* particionados e consequentemente minimiza-se a função objetivo.

Em um efeito simultâneo, a solução encontrada pelo HSCM povocou dois agrupamentos

entre duas distribuições sintéticas em um único *cluster*. Evidentemente isso produz um aumento da contribuição das observações asssociadas a estas das distribuições sintéticas ao valor da função objetivo.

Porém, mais uma vez, esse efeito foi amplamente compensado pela ganho decorrente do triplo particionamento das observações da distribuição sintética D10, fato compravado pelo decrécimo de 27.55% no valor de H_{best} em relação a H_{sint} apresentado na tabela 4.3.

Este foi o segundo melhor resultado obtido pelo HSCM, praticamente o mesmo do Problema Teste T04. Neste sentido deve-se notar que no Problema Teste T09 possui uma configuração especial na qual existem dois pares de distribuições sintéticas perto o suficiente para compensar o triplo particonamento.

Outro aspecto que deve ser obeservado aqui, assim como no teste T08, a coluna *delta obs* mostra que a solução obtida pelo HSCM produziu duas perfeitas colapsações das observações associadas às distribuições D1 , D2, D3 e D4 em dois *clusters* : h1 e h2.

Também podemos observar que oparticionamento triplo da distribuição sintética D9 nos três *clusters* h8 , h9 e h10 foi tão perfeito quanto o observado no problema Teste T08.

Em virtude deste resultado excepcional, este problema teste foi escolhido para uma análise mais profunda do seu processo de convergência para esta solução.

Análise da convergência dos parâmetros

Quatro dos dez pontos de partida (*starting points*) convergiram para o melhor resultado H_{best}. A tabela 4.11 mostra o processo de convergência particular ocorrido no processamento do ponto de partida 3, um dos quatro melhores.

Nesta tabela, a primeira coluna k representa a iteração do algoritmo. A segunda coluna $H(X_k)$ apresenta o valor da homogeniedade calculada pelo critério da soma mínimos quadrados para os agrupamento das observações s_j em torno do conjunto de centróides $X_k = \{x_1^k, \ldots, x_q^k\}$ encontrados na iteração k .

$$H(X_k) = \sum_{j=1}^{m} z_j^2 \tag{4.7}$$

$$z_j = \min_{x_i \in X_k} ||s_j - x_i||_2 \tag{4.8}$$

As demais colunas apresentam o valor dos principais parâmetros do método na iteração:

$\gamma^k, \tau^k, \varepsilon^k$, e a variação (δH) entre a função de homogeniedade dada por:

$$\delta H \quad = \quad H(X_{k+1}) - H(X_k) \tag{4.9}$$

k	$H(X_k)$	tal(τ^k)	gamma (γ^k)	eps(ε^k)	δH
1	8.32024869e+02	3.50415724e-02	3.50415724e-04	1.40166290e-01	
2	8.35946559e+02	8.76039310e-03	8.76039310e-05	3.50415724e-02	3.92170e+00
3	8.32349494e+02	2.19009828e-03	2.19009828e-05	8.76039310e-03	-3.59710e+00
4	8.32070971e+02	5.47524569e-04	5.47524569e-06	2.19009828e-03	-2.78520e-01
5	8.31921775e+02	1.36881142e-04	1.36881142e-06	5.47524569e-04	-1.49200e-01
6	8.31849196e+02	3.42202856e-05	3.42202856e-07	1.36881142e-04	-7.25790e-02
7	8.31848937e+02	8.55507139e-06	8.55507139e-08	3.42202856e-05	-2.58800e-04
8	8.31848916e+02	2.13876785e-06	2.13876785e-08	8.55507139e-06	-2.10190e-05
9	8.31848898e+02	5.34691962e-07	5.34691962e-09	2.13876785e-06	-1.81760e-05
10	8.31848898e+02	1.33672990e-07	1.33672990e-09	5.34691962e-07	-3.14830e-07

Tabela 4.11: Convergência Ponto 3 do Teste T09

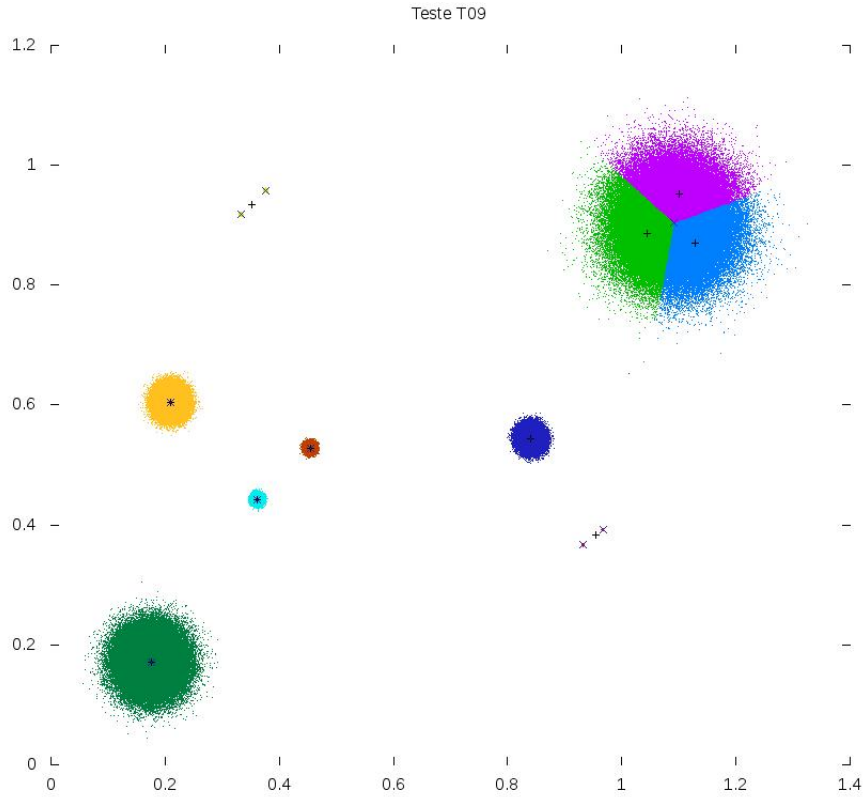

Figura 4.10: Resultados HSCM T09 - 10 *clusters*

Análise do Critério de Parada

No processamento dos problemas teste foi fixado *à priori* como critério de parada, a realização de no máximo 10 iterações. Outros critérios mais adequados podem ser utilizados como a variação do valor da função objetivo ou ainda da variação no posicionamento dos centróides.

Específicamente para este Problema Teste a adoção de um critério de parada para variação de $H(X_k+1)-H(X_k) < 10^{-3}$ interromperia o processamento em 3.43 segundos.

A análise dos parâmetros $\gamma^k, \tau^k, \varepsilon^k$ revela efeito do parâmetro ρ incializado com o valor de 10^{-2}. Os parâmetros se relacionam entre si, segundo as seguintes relações:

$$\gamma^{k+1} = \rho\gamma^k \tag{4.10}$$

$$\tau^{k+1} = \rho\tau^k \tag{4.11}$$

$$\varepsilon^{k+1} = \rho\varepsilon^k \tag{4.12}$$

A figura 4.10 mostra a representação gráfica da solução do problema teste T08 As médias de cada distribuição normal estão assinaladas na figura pelo símbolo "+". Os parâmetros (μ_i, σ_i)

destas distribuições encontram-se na tabela 4.2.

As figura mostra os limites de cada uma das distribuições sintéticas por essas estarem visivelmente separadas.

O processamento do agrupamento pelo HSCM foi realizado com o número de grupos $q = 10$ igual ao número de distribuições sintéticas.

Cada *cluster* é representado por uma cor diferente na figura. Os seus centróides estão assinalados pelo símbolo "×".

Quando a média e o centróide coincidem ocorre a fusão do símbolo "+"com o símbolo "×"transformando-se em "*"

5 *Conclusões*

A presente dissertação de mestrado teve por objetivo analisar o impacto da explosiva taxa de crescimento do volume dos dados sobre as metodologias de *Data Mining* e apresentar algumas abordagens emergentes para solução deste problema no campo de *clustering*.

Na introdução apresentamos a evolução do fenômeno das bases de dados gigantes, sua abrangência e consequências no contexto do *Data Mining*. Apresentamos ainda os desafios e as oportunidades que este campo de pesquisa oferece. Por fim, justificamos a escolha do tema devido ao seu alinhamento com o interesse estratégico da Dataprev neste assunto.

A seguir, fizemos uma revisão sucinta do tema *Data Mining* fundamentada no artigo *The Top Ten Algorithms in Data Mining* [25] que cobre um vasto espectro das técnicas associadas a este tema.

Cabe ressaltar que este artigo foi realizado com a contribuição de eminentes pesquisadores, reconhecidos como autoridades em seu campo de pesquisa, selecionados entre os ganhadores dos principais prêmios internacionais sobre o tema.

Para estudar o comportamento de um algoritmo no contexto de base de dados gigantes, tomamos a decisão de escolher um algoritmo associado a um dos principais campos de pesquisa do *Data Mining*: o problema de *clustering*.

Entre os algoritmos emergentes neste campo, escolhemos o HSCM , um algoritmo alternativo ao consagrado *k-means* , porque ambos tentam resolver exatamente o mesmo problema matemático para obter uma solução para o problema de *clustering*.

Uma vez que as maiores bases de dados encontradas na literatura para problemas de *clustering* situam-se na ordem de grandeza de 10^3 observações , resolvemos construir problemas teste sintéticos na ordem de 10^6 observações para simular problemas caracteríticos de bases de dados gigantes.

Realizamos um conjunto de experimentos computacionais para avaliar o comportamento do HSCM no contexto destes problemas teste sintéticos gigantes. Apresentamos e analisamos

estes resultados sob os apectos de acurácia (precisão dos resultados), consistência , eficiência (tempo de processamento). A tabela 6.1 apresentada no Anexo A, exibe um comparativo entre estes resultados e os obtidos pelo *k-Means* nas mesmas condições.

Esta dissertação contribuiu com a criação de algumas ferramentas úteis para as futuras pesquisas com este algoritmo. Entre as ferramentas desenvolvidas para este trabalho podemos destacar o módulo de integração entre o *log* (registro das variáveis de contexto do processamento) do HSCM e ferramentas de analise e visualização dos resultados.

Entre as funcionalidades deste módulo de integração temos a transformação dos *logs* , que são arquivos texto não estruturados em arquivos estruturados prontos para carga no R, MATLAB e/ou Excell. Todos os gráficos e tabelas apresentados na análise dos resultados desta dissertação foram processados por esta ferramenta.

Outra funcionalidade importante deste módulo é a função de classificação que atribui aos arquivos de entrada do HSCM a classe que cada observação pertence segundo a métrica do centróide mais próximo.

Pelos resultados obtidos podemos considerar que o HSCM teve uma excelentente performance segundo todos critérios avaliados. Podemos concluir que o HSCM mostrou-se uma ferramenta extremamente adequada a resolver problemas de clustering em bases de dados gigantes.

Em suma, acreditamos que podemos usar a metodologia do HSCM na solução de problemas de *clustering* em diversos contextos e em especial na Previdência Social.

Referências Bibliográficas

[1] Agrawal R., Srikant R. 1994. Fast algorithms for mining association rules.*Proceedings of the 20thVLDB conference* , pp 487?499.

[2] BAGIROV, Adil M. 2008. Modified global -means algorithm for minimum sum-of-squares clustering problems. *Pattern Recognition*, 2008, vol. 41, no. 10, p. 3192-3199.

[3] Barroso, L. A., Dean,J. e Hölzle, U. *Web search for a planet: The Google cluster architecture.* IEEE Micro, 23(2):22.28, 2003.

[4] Bashashati, A. and Brinkman, R.R. 2009. A survey of flow cytometry data analysis methodss. *Advances in bioinformatics*, 2009, Hindawi Publishing Corporation.

[5] BREIMAN L., FRIEDMAN J. H., Olshen R.; C., Stone. 1983. Classification and regression trees. 1983, Wadsworth, Belmont,California.

[6] Brin S., Page L. 1998. The anatomy of a large-scale hypertextual Web Search Engine. *Comput Networks* 30(1:7):107-117

[7] Cuzzocrea, A. e Gaber, M. 2012. Data science and distributed intelligence: recent developments and future insights.,In: *Proceedings of the 6th International Symposium on Intelligent Distributed Computing - IDC* 2012, 24-26 September, 2012, Calabria, Italy

[8] Davenport, T. H. *Competing on analytics. harvard business review*, 84(1), 98.(2006)

[9] Dean,J. e Ghemawat,S. 2004 MapReduce: Simplified Data Processing on Large Clusters *OSDI2004*

[10] Dempster, A.P., Laird, N.M. e Rubin, D.B. 1977 Maximum likelihood from in complete data via the EM algorithm *Journal of the Royal Statistical Society* Serie B, 39:1-38, 1977.

[11] Ellis, B. , Haaland, P. , Le Meur,N. et Gopalakrishnan,N. flowCore: Basic structures for flow cytometry data.. *R package version 1.22.3.* , 2012.

[12] Fayyad, U., Shapiro, G. P. et Smyth, P. From Data Mining to Knowledge Discovery in Databases. *AI MAGAZINE Fall 1996* 37-51 , 1996.

[13] Freund, Y. et Schapire,R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119-139, 1997.

[14] Gentleman, R. , Carey,J., et all Bioconductor: Open software development for computational biology and bioinformatics R. *Genome Biology*, Vol. 5, R80, 2004.

[15] Ghoting A. , Kambadur P., Pednault E. P. D., e Kannan R. 2011. *Nimble: a toolkit for the implementation of parallel data mining and machine learning algorithms on mapreduce.* em KDD2011, páginas 334-342, 2011.

[16] HAN, J. , KAMBER, M. e PEI,J. 2011. Data Mining: Concepts and Techniques. 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

[17] HAN,J., PEI, J e YIN, Y. 2000. Mining frequent patterns without candidate generation. em *Proceedings of ACM SIGMOD international conference on management of data.* , paginas 1-12 3rd. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. 2000.

[18] Hartingan, J.A. 1975 Clustering Algorithms Wiley, New York, 1975.

[19] Hastie,T. e Tibshirani,R. 1966. *Discriminant adaptive nearest neighbor classification* em *IEEE Trans. Pattern Anal. Mach. Intell.*, páginas 607-616, 1996.

[20] HILBERT, M., et LOPEZ , P. *How to measure the world?s technological capacity to communicate, store and compute information? Part I: Results and scope. International Journal of Communication 6* pg: 956-979 2012.

[21] McQueen, J. 1967 *Some methods for classification and analysis of mutivariate observations* Proc. 5th Berkeley Symp. Math., Statistics and Probability, 1, pp. 281-296, 1967

[22] McLachlan, G.J. e Peel, D. 2000 Finite Mixture Models Wiley, New York, 2000.

[23] Papadimitriou S., Sun J. e Yan R. *Large-scale data mining: Mapreduce and beyond* Tutorial em KDD10, Julho 2010

[24] Quinlan, J. R. C4.5: Programs forMachine Learning. Morgan Kaufmann, 1993.

[25] WU, X. et all. top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 2007, Springer-Verlag New York, Inc., New York, NY, USA, 14, no. 1, p. 1-37.

[26] R Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/, 2012

[27] Wu, Xindong and Kumar, Vipin The Top Ten Algorithms in Data Mining Ed.1st, Boca Raton FL Chapman & Hall/CRC

[28] XAVIER, Adilson Elias. The hyperbolic smoothing clustering method.*P*attern Recognition, 2010, vol. 43, no. 3, p. 731-737.

[29] Xu,X. , Kriegel,H.P.,Kriegel,S. e Ester,M. A distribution-based clustering algorithm for mining in large spatial databases. *Data Engineering, 1998. Proceedings., 14th International Conference on* p324 - 331 IEEE,1998

6 Anexo

Anexo A - Tabela Comparativa k-Means X HSCM

Teste	q	H_{kMeans}	H_{HSCM}	% $\Delta H_{kMeans-HSCM}$	T_{kMeans}	T_{HSCM}	$T_{kMeans/HSCM}$
T01	2	9105.12	9105.12	0.00%	5.48	1.41	3.89
T02	3	9510.99	9510.99	0.00%	7.18	1.81	3.97
T03	4	944.05	944.05	0.00%	7.12	2.18	3.27
T04	5	9415.00	6785.75	0.00%	16.54	4.20	3.94
T05	6	5302.48	5101.52	0.00%	11.61	4.46	2.60
T06	7	7682.60	5009.09	35.00%	10.74	4.52	2.38
T07	8	1425.77	1178.42	17.00%	11.43	4.35	2.63
T08	9	1956.88	1956.88	0.00%	11.79	4.63	2.55
T09	10	1151.74	831.85	28.00%	10.78	4.92	2.19

Tabela 6.1: Tabela Comparativa k-Means x HSCM com dados sintéticos T01 a T09

Os resultados do *k-Means* foram obtidos utilizando-se a implementação disponibilizada pelo pacote *stats* do projeto **R** [26] baseado no artigo de Hartigan [18] . Foram utilizados as mesmas condições de execução do HSCM :

- dez pontos de partida inicial ($nstart = 10$)
- critério de parada de no máximo 10 iterações ($iter.max = 10$)