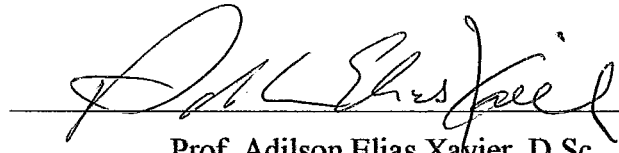
MINERAÇÃO DE DADOS VIA MÁQUINA DE VETORES SUPORTE COM SUAVIZAÇÃO HIPERBÓLICA

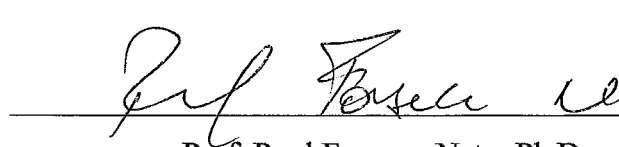
Patrícia Curvelo Rodrigues

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA 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.

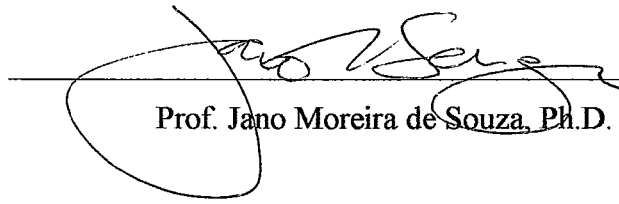
Aprovada por:



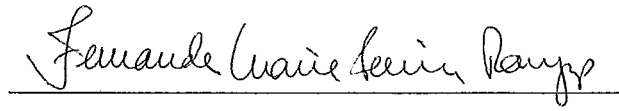
Prof. Adilson Elias Xavier, D.Sc.



Prof. Raul Fonseca Neto, Ph.D.



Prof. Jano Moreira de Souza, Ph.D.



Prof. Fernanda Maria Pereira Raupp, D.Sc

RIO DE JANEIRO, RJ - BRASIL

FEVEREIRO DE 2008

RODRIGUES, PATRICIA CURVELO

Mineração de Dados via Máquina de Vetores Suporte com Suavização Hiperbólica [Rio de Janeiro] 2008

X, 81 p. 29,7 cm (COPPE/UFRJ, M.Sc., Engenharia de Sistemas e Computação, 2008)

Dissertação – Universidade Federal do Rio de Janeiro, COPPE

1. Mineração de dados
2. Suavização Hiperbólica
3. SVM

I. COPPE/UFRJ II. Título (série)

Dedicatória

Dedico esta tese aos meus pais e ao
meu esposo Victor.

Agradecimentos

Quando era adolescente, ouvi de uma grande amiga que somos como caixas d'água. Damos o que recebemos e temos sempre que nos abastecer para continuar transmitindo às pessoas ao redor um pouco do que somos. É dessas, e para essas, pessoas que abastecem a minha caixa d'água que vou falar agora.

Deus! Minha principal fonte de energia e esperança. Agradeço por me dar forças e estar sempre colocando pessoas do bem em meu caminho.

Meus pais. Meus exemplos de força, luta, companheirismo, carinho e, acima de tudo, moral. Além de terem me criado com muito carinho e de nunca terem deixado faltar nada que eu precisasse.

Meu marido. Está comigo em todos os momentos. Como diriam os Paralamas, "*aonde quer que eu vá, levo você no olhar*". Amigo, companheiro no mais amplo sentido da palavra. Obrigada pelo carinho e por estar sempre por perto.

Meus irmãos. Apóiam-me e ajudam até mesmo sem saber, são os responsáveis por vários sorrisos e momentos de descontração. Obrigada pelo carinho.

Minhas avós. Já não as tenho mais por perto, mas são os meus exemplos de generosidade e perseverança. O curto tempo que estivemos juntas foi suficiente para aprender que ser do bem e fazer o bem é sempre o melhor caminho.

Adilson Elias Xavier. Já não sei nem se o chamo mais de professor, pois no decorrer desse trabalho se tornou um grande amigo. Entretanto, será sempre meu mestre. Obrigada pela amizade, carinho e dedicação na orientação deste trabalho. Por sua inteira disposição e paciência para transmitir todo o conhecimento necessário para o desenvolvimento desta dissertação de mestrado.

Meus amigos. Não posso nem citar o nome e a importância de cada um deles, pois não haveria espaço nessa dissertação. Dizem que "amigos são a família que podemos escolher" e esses fazem jus ao ditado popular. Obrigada pelo apoio incondicional e pelos momentos juntos.

Fundação COPPETEC, na pessoa dos professores Jano e Blaschek. Obrigada por abrirem as portas quando precisei e por confiarem em meu trabalho.

Fundação CAPES. Obrigada pela bolsa concedida para execução desse trabalho.

Professores Raul, Jano e Fernanda. Obrigada por terem aceitado tão prontamente o convite para participação na Banca de Defesa de Tese.

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

MINERAÇÃO DE DADOS VIA MÁQUINA DE VETORES SUPORTE COM SUAVIZAÇÃO HIPERBÓLICA

Patrícia Curvelo Rodrigues

Fevereiro/2008

Orientador: Adilson Elias Xavier

Programa: Engenharia de Sistemas e Computação

Este trabalho é destinado a mostrar o processo de Mineração de Dados com enfoque no principal processo de mineração que é a classificação dos dados. Mostra todas as etapas do processo de mineração, apresentando em cada uma delas as principais características e técnicas aplicadas. Dentro do processo de classificação, é apresentada uma nova abordagem para a resolução do problema linear de Máquina de Vetor Suporte (SVM) como parte de um fluxo no processo de Mineração de Dados. O modelo matemático considerado conduz a uma formulação que tem uma característica significante, de ser não-diferenciável. Para superar esta dificuldade, o método de resolução proposto adota uma estratégia de suavização usando uma função suavizadora especial pertencente à classe de funções C^{∞}, a Suavização Hiperbólica.

A solução final é obtida através da resolução de uma seqüência de subproblemas de otimização diferenciáveis irrestritos, definidos em um espaço com dimensão pequena, que gradativamente se aproximam do problema original. A utilização dessa técnica, denominada Suavização Hiperbólica, permite superar as principais dificuldades presentes no problema original. Um algoritmo simplificado contendo somente o essencial do método é apresentado e os resultados computacionais são comparados ao mais conhecido software de classificação via SVM, o *SVMLight.*

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

DATA MINING BY SUPPORT VECTOR MACHINE CLASSIFICATION APPROACH WITH HYPERBOLIC SMOOTHING

Patrícia Curvelo Rodrigues

February/2008

Advisor: Adilson Elias Xavier

Department: Systems Engineering and Computer

This work is intended to show the Data Mining procedure with focus on the main task of mining which is the data classification. It is shown all stages of the mining procedure, presenting for each one the key features and techniques applied. In the classification task, a new approach to solving the problem of linear Support Vector Machine (SVM). The mathematical modeling of this problem leads to a formulation which has the significant characteristic of being non-differentiable. In order to overcome these difficulties, the resolution method proposed adopts a smoothing strategy using a special C^{∞} differentiable class function, called Hyperbolic Smoothing.

The final solution is obtained by solving a sequence of low dimension differentiable unconstrained optimization sub problems, which gradually approach the original problem. The use of this technique, called Hyperbolic Smoothing, allows the main difficulties presented by the original problem to be overcome. A simplified algorithm containing only the essential of the method is presented and the results are compared to the one of the best know software for SVM classification, the *SVMLight*.

Sumário

Índice de Figuras

Índice de Tabelas

Capítulo 1 – Introdução

A quantidade de informações disponíveis atualmente nos meios de comunicação aguça a curiosidade humana pela captura do conhecimento intrínseco nesse emaranhado de dados. Entretanto, o ser humano, ao se ver cercado de tantas alternativas e opções, não é capaz de fazer uma avaliação precisa sobre essas informações e adquirir o conhecimento desejado. Isso ocorre porque a mente humana não é capaz de analisar todos os dados que estão ao seu redor com o detalhamento necessário, em particular, aqueles com grande número de atributos.

Buscando solucionar esse problema, várias propostas têm sido desenvolvidas, na tentativa de auxiliar o ser humano na tomada de decisão em ambientes cercados de incertezas e imprecisões. Dentre essas propostas, a que tem sido mais explorada é a Mineração de Dados.

A Mineração de Dados é uma tecnologia nova e com acepção ainda não plenamente consolidada. Diz respeito a um conjunto de técnicas e procedimentos intrínsecos à extração de conhecimento em grandes volumes de dados. A terminologia Mineração de Dados está inspirada nas peripécias intrínsecas às atividades de extração de minérios, mineração. Assim como em um processo de mineração de ouro, um garimpeiro procura o ouro oculto em meio a terra, na Mineração de Dados o analista igualmente procura algum tipo de conhecimento oculto a partir de um "emaranhado" de dados.

Nos primeiros sistemas computacionais, as duas maiores preocupações eram com o armazenamento das informações e com a execução do sistema (portabilidade). Entretanto, o maior desses problemas já foi sanado, o espaço em disco. O que se tem atualmente é uma capacidade praticamente "infinita" de armazenamento de dados.

Esse aumento na capacidade de armazenamento, aliado ao surgimento dos sistemas de bancos de dados, e ao aumento do poder de processamento, proporcionam o surgimento de questões mais complexas. Por exemplo, gerentes de empresas querem saber como os dados armazenados de seus clientes podem ser úteis para o seu negócio. Não tem muito sentido ficar armazenando informações que não serão analisadas e utilizadas para aumentar os ganhos dentro da empresa. A pergunta é: como obter conhecimento a partir de um enorme emaranhado de dados?

As respostas para a maioria das questões estão embutidas nas bases de dados das próprias empresas. Entretanto, não se encontram explícitas. Os sistemas de gerenciamento de bases de dados não permitem uma exploração dos dados, não fazem nenhum tipo de resumo de forma a tornar clara a informação contida no mesmo. Além disso, não administram de forma adequada modelos de bases de dados muito grandes.

Promover essas capacidades de extração da informação é a meta da emergente área de pesquisa Descobrimento de Conhecimento em Base de Dados (BRADLEY *et al.*, 1998), mais precisamente da etapa Mineração de Dados (MD). Essa área está rapidamente envolvendo áreas de pesquisa multidisciplinares, incluindo estatística, banco de dados, reconhecimento de padrões, inteligência artificial, otimização e visualização, entre outras. O objetivo principal dessas pesquisas é encontrar informações úteis e que normalmente não estão visíveis em grandes bases de dados.

O processo de Descobrimento em Bases de Dados (do inglês, *Knowledge Discovery in Databases,* KDD) envolve uma série de etapas, como pode ser visto na Figura 1.1. Fazendo uma supersíntese pode-se fazer a seguinte explicação dessa figura que será detalhada no capítulo 2.

A partir de uma, ou mais, bases de dados é feita a seleção dos dados a serem estudados. Nessa etapa, alguns dados são extraídos e aglomerados formando uma única massa de dados. Em seguida, é feita uma limpeza desse conjunto, ou seja, são eliminadas as informações que se encontram de forma inadequada ao processo, todos os dados com algum tipo de ruído é eliminado ou tratado.

Com o conjunto livre de incoerências, é feita uma transformação sobre os dados. Nessa etapa, os dados são transformados em vetores ou em alguma estrutura previamente definida, de forma que possibilite a etapa de mineração propriamente dita. Na mineração são utilizados algoritmos de classificação, agrupamento e seleção, dentre outros. Assim, os padrões são retirados e em seguida avaliados segundo um critério previamente escolhido. No término desse processo, o padrão identificado é mostrado ao usuário da forma mais clara possível.

Embora o processo de KDD seja aparentemente simples, cada uma de suas etapas não é executada como um simples "passo de mágica". Por isso, é importante ressaltar a importância do analista durante todo o processo de KDD. A presença de um especialista no assunto possui um papel fundamental na qualidade e precisão dos resultados. Um estudo mal direcionado em qualquer etapa do processo pode conduzir a

análises erradas ou com pouco grau de relevância para o usuário. Cabe ao analista supervisionar todo o processo e garantir que a avaliação final seja positiva.

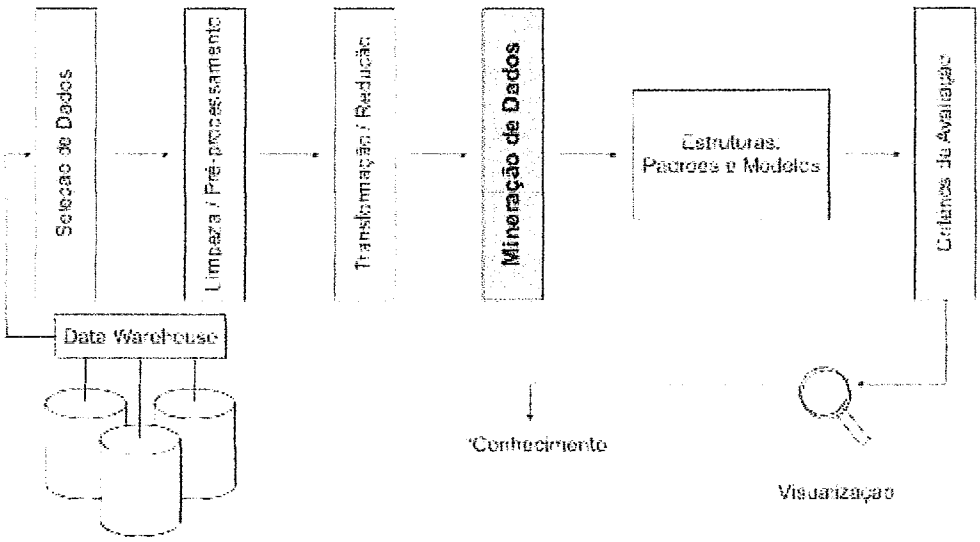


Figura 1.1 - Etapas do processo de KDD (BRADLEY *et al.*, 1998)

Dentre os passos a serem seguidos na etapa de Mineração de Dados destaca-se a extração do conhecimento. Essa extração pode ser feita de várias formas: através de algoritmos de classificação, regressão e agrupamento, dentre outros. O objetivo desta dissertação é discutir uma nova metodologia para se aplicar à MD, uma técnica de classificação de dados segundo o enfoque SVM. Essa técnica é uma das inúmeras técnicas usadas no escopo de MD.

Os algoritmos de classificação estão intimamente associados ao conceito de aprendizado. A partir da incorporação desse conceito, os algoritmos de classificação se tornam capazes de aperfeiçoar seu desempenho em sua tarefa precípoa, no caso, a classificação. Para isso, é utilizado um conjunto de dados previamente escolhido, denominado conjunto de treinamento, e uma função de aprendizado. Os parâmetros de ajuste da função de aprendizado escolhida são manipulados de forma a torná-la a mais adequada possível ao conjunto de treinamento. Dessa forma, é gerado um classificador que tenta extrair informações estruturais não explícitas sobre diversos aspectos implícitos no conjunto de dados.

Uma analogia simplificada a esse processo pode ser feita a partir de um cliente de supermercado. A primeira vez que o cliente compra determinado produto, ele possui

apenas as informações obtidas através de propagandas ou indicações de pessoas conhecidas. Já na segunda, o cliente conta com as informações que ele recebeu e com a sua própria experiência. Com isso, após vários testes de produtos diferentes, o cliente é capaz de classificar se uma determinada mercadoria é boa ou ruim. Assim, ele se torna capaz de avaliar uma mercadoria desconhecida a partir das suas experiências e das características de produtos similares que ele julga mais importantes.

Particularmente, em problemas de classificação relacionados a um conjunto de treinamento, há um processo inicial, onde se busca inferir uma hipótese caracterizada pelo projeto de um classificador. Essa hipótese é construída representando os dados do conjunto de entrada em vetores e treinando o classificador para que ele identifique corretamente todos, ou a maioria dos dados do conjunto de treinamento. Posteriormente, o classificador pode estabelecer uma categorização para uma nova amostra, desde que essa seja representada de forma vetorial como os dados do conjunto de treinamento.

A Figura 1.2 ilustra, de forma simplificada, o processo de aprendizado do classificador. Nesse exemplo, pode ser observado que existe um conjunto de classes distintas do mundo real. O objetivo, como dito anteriormente, é representar cada dado dessas classes de forma vetorial, passar por uma etapa de treinamento e construir o classificador. A etapa de pré-processamento transforma os dados em uma forma vetorial para que os mesmos possam ser interpretados pelo algoritmo de classificação. Já a etapa de treinamento tem a função de "ensinar" o classificador para que ele "aprenda" sobre as características de cada classe do conjunto de entrada.

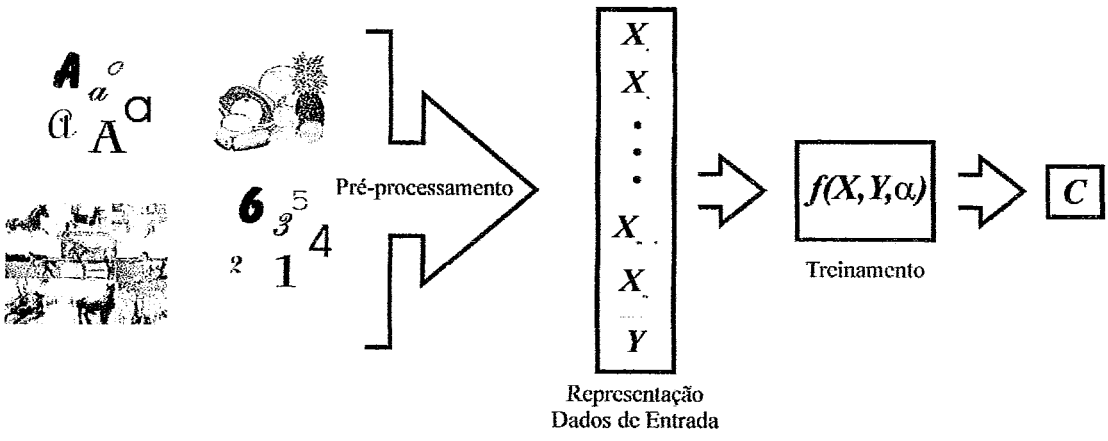


Figura 1.2 - Representação da etapa de aprendizado de um classificador.

Na maioria das vezes, esses classificadores são treinados para atribuir os dados a duas classes diferentes. Assim, cada classificador é representado por uma função discriminante que, na forma mais simples, será um hiperplano, quando os dados forem linearmente separáveis, ou uma outra função, para dados não linearmente separáveis.

Uma das estratégias de maior sucesso no equacionamento de problemas de classificação é a denominada Máquina de Vetor Suporte, mais conhecida por sua denominação em inglês *Support Vector Machine* (SVM) (VAPNIK, 1995).

Um dos softwares mais utilizados na resolução de problemas de classificação, segundo a abordagem SVM, é o *SVMLight* (JOACHIMS, 1998). No decorrer do aprimoramento desse software, também foram desenvolvidas técnicas para redução do conjunto de treinamento, de forma a reduzir o tempo de processamento gasto pelo classificador durante a fase de treinamento. Assim, após dez anos, esse software tornou-se um dos mais eficientes na resolução de problemas SVM dentre os softwares conhecidos na literatura.

Neste trabalho será apresentada uma técnica, baseada em uma nova metodologia, para resolução do problema de classificação. A nova metodologia é baseada nos conceitos de Suavização Hiperbólica (XAVIER, 1993) e Penalização Hiperbólica (XAVIER, 1982). Dessa forma, espera-se apresentar uma nova alternativa metodológica no processo de classificação dentro do escopo de Mineração de Dados.

1.1 – Seqüência da apresentação

No presente trabalho, considera-se a resolução do problema de classificação através da abordagem de Máquina de Vetor Suporte, ou *Support Vector Machine* (SVM). Essa abordagem é relativamente nova na literatura, sendo a sua proposição original apresentada por Vapnik em 1995 (VAPNIK, 1995).

Desde então, a comunidade científica tem dedicado uma atenção bastante expressiva ao tema. Esse grande interesse sobre o problema SVM decorre do mais amplo escopo de suas aplicações práticas, podendo ser aplicado em diversas áreas na solução de problemas distintos, entre eles a Mineração de Dados.

A formulação matemática do problema SVM apresenta algumas particularidades como a não linearidade, não convexidade e não diferenciabilidade. Essas características adversas dificultam a utilização de métodos ortodoxos de solução de problemas de otimização com restrição na resolução desse problema.

A técnica de suavização hiperbólica será adotada como uma alternativa para a solução do problema SVM. Essa técnica corresponde a um desdobramento direto do método da penalização hiperbólica, destinado à solução do problema geral de programação não linear com restrições, originalmente apresentado por Xavier em 1982 (XAVIER, 1982).

A suavização hiperbólica tem sido usada para a solução de diversos problemas de programação matemática não diferenciável. Primeiramente foi utilizado para a resolução de um problema de calibração automática de modelos hidrológicos, como apresentado em (DIB, 1994) e em (SILVA *et al.*, 1990). Em seguida, foi adotada para resolver um problema de controle elétrico, conforme apresentado em (MOTA *et al.*, 1992), e para a minimização de funções definidas por mais de uma cláusula, conforme em (XAVIER, 1993).

Como os resultados obtidos inicialmente foram de fato satisfatórios, essa abordagem foi em seguida utilizada para a resolução de um problema de grande importância teórica e prática, o problema *min-max*. A descrição detalhada desse problema, bem como sua resolução através da técnica de suavização hiperbólica são encontrados em (CHAVES, 1987) e em (CHAVES *et al.*, 1998).

Uma síntese desse conjunto de aplicações é apresentada em (SANTOS, 1997). Mais recentemente, essa técnica tem sido utilizada na resolução de problemas de recobrimento, simples e múltiplo, de uma região plana por círculos, conforme apresentado em (XAVIER, 2000), (XAVIER *et al.*, 2003), (XAVIER, 2005) e (BRITO, 2004). As aplicações mais recentes dessa abordagem consideram a resolução de problemas de agrupamento (*Clustering*), em (XAVIER, 2005), e o de classificação segundo critério de máquina de vetor suporte, em (XAVIER *et al.*, 2006).

Na técnica de suavização hiperbólica, em todos os problemas acima citados, a solução é obtida através da resolução de uma seqüência infinita de problemas continuamente diferenciáveis, classe C^{∞}, que gradativamente se aproximam do problema original. Registra-se que o desempenho computacional dessa técnica, frente a todos esses problemas, sempre obteve êxito.

Neste trabalho de dissertação, considera-se, em particular, a extensão e o aprimoramento do uso da abordagem da suavização hiperbólica para a resolução do problema SVM, primeiramente utilizada no artigo (XAVIER *et al.*, 2006), com aplicação no problema de Mineração de Dados. Para melhor descrever essa técnica esta dissertação está organizada de acordo com a seguinte seqüência. No capítulo 2 é feita

uma descrição do processo de Mineração de Dados como um todo. Cada uma de suas etapas é detalhada e é mostrado em cada uma delas quais as principais técnicas e ferramentas a serem utilizadas. Dentre elas, a ferramenta de classificação na etapa de extração de padrões, que é o foco dessa dissertação.

No capítulo 3 é dada a definição de classificação, bem como a descrição de algumas técnicas utilizadas na resolução desses problemas. Além disso, é feita uma pequena revisão bibliográfica sobre diversos métodos de classificação, sendo a formulação matemática desses problemas definida de maneira formal.

No capítulo 4 é proposta uma nova metodologia para a resolução do problema SVM denominada SHSVM. Para tal, é feita uma descrição matemática formal do problema SVM e suas principais características.

No capítulo 5 é feita uma análise detalhada da implementação da metodologia proposta. Além disso, são apresentados os experimentos computacionais desenvolvidos e os correspondentes resultados obtidos. Esses resultados são comparados aos obtidos pelo *SVMLight*.

Finalmente, no capítulo 6 são apresentadas as conclusões e sugestões para futuros trabalhos.

Capítulo 2 – Mineração de Dados

Com a evolução dos sistemas de gerenciamento de bancos de dados, com o aumento da velocidade dos computadores e do volume de dados armazenados e com a facilidade atual de se gerar uma grande massa de dados em um curto intervalo de tempo, veio o interesse em se obter informações extra sobre os dados armazenados. A Mineração de Dados (MD) é uma ferramenta que auxilia na extração de informações adicionais contidas nos bancos de dados. Permite aos pesquisadores e analistas de mercado fazer análises, interpretações e projeções sobre os dados de uma forma mais abrangente e correta.

Duas das aplicações mais conhecidas do processo MD são estudos de padrões de consumo em lojas da cadeia de hipermercados Wal-Mart e o estudo feito para o vestibular da PUC-Rio (WIKIPÉDIA, 2007). O primeiro caso procurou eventuais relações entre o volume de vendas nos hipermercados da rede e os dias da semana. O software apontou que, às sextas-feiras, as vendas de cervejas cresciam na mesma proporção que as de fraldas. Uma investigação mais detalhada revelou uma relação oculta nos dados. Essa relação inferiu que ao comprar fraldas para seus bebês, os pais aproveitavam para abastecer as reservas de cerveja para o final de semana.

No segundo caso, constatou-se que se o candidato é do sexo feminino, trabalha e teve aprovação com boas notas no vestibular, então não efetivava a matrícula. Analisando mais a fundo, foi possivel verificar a regra. De acordo com os costumes do Rio de Janeiro, uma mulher em idade de vestibular, se trabalha é porque precisa, e neste caso deve ter feito inscrição para ingressar na universidade pública gratuita. Se teve boas notas provavelmente foi aprovada na universidade pública onde efetivará matrícula.

Uma pergunta do tipo: "qual produto de alta lucratividade venderia mais com a promoção de um item de baixa lucratividade, analisando os dados dos dez últimos anos de vendas?", não pode ser respondida através de um simples comando *SQL* ou através de uma elaborada planilha de dados. Entretanto, pode ser de fundamental importância para uma determinada empresa. A partir desse problema foi criada uma nova área denominada Descoberta de Conhecimento em Base de Dados ou simplesmente KDD (*Knowledge Discovery in DataBases*).

Segundo Golschimdt e Passos (GOLSCHIMDT *et al.*, 2005; GOLSCHIMIDT *et al.*, 2005) "O termo KDD foi formalizado em 1989 em referência ao amplo conceito de procurar conhecimento a partir de bases de dados". Segundo Fayyad (BRADLEY *et al.*, 1998) "KDD é um processo, de várias etapas, não trivial, interativo e iterativo, para a identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados".

Geralmente, as pessoas tendem a chamar todo o processo de KDD de Mineração de Dados ou *Data Mining*. Entretanto, a realização da Mineração de Dados (GOLSCHIMDT *et al.*) é apenas uma etapa desse processo, possibilitando a extração de dados complexos da base de dados.

Uma das tecnologias de suporte à Mineração de Dados (GOLSCHIMDT *et al.*) que tem sido mais utilizada é o *Data Warehousing*. O *Data Warehousing* é um processo, não um produto, para montar e gerenciar um conjunto de repositórios de dados a partir de várias fontes com o propósito de se ter uma visão detalhada e singular de parte ou do todo de um negócio (REZENDE *et al.*, 2003).

A realização de *Data Warehousing* é considerada um dos primeiros passos para tornar factível a análise de grande quantidade de dados no apoio ao processo decisório. O objetivo básico é criar um repositório conhecido por *Data Warehouse* (DW)[1] que contenha dados limpos, agregados e consolidados que possam ser analisados por ferramentas OLAP (*On-Line Analytical Processing*). As ferramentas utilizadas para analisar um DW, normalmente, são orientadas às consultas, ou seja, são definidas pelos usuários, os quais possuem hipóteses que gostariam de comprovar, ou simplesmente, executar consultas aleatórias. Essa abordagem pode impedir que dados realmente significativos que estejam escondidos na massa de dados possam ser encontrados, pois é humanamente impossível para um usuário ter condições de imaginar ou executar todas as combinações e relações possíveis em um grande volume de dados. Dessa forma, torna-se necessária a criação de uma ferramenta que seja capaz de analisar e extrair de forma automática (ou semi-automática) novos conhecimentos a partir de um grande repositório de dados (BRADLEY *et al.*, 1998).

---

[1] Um DW representa uma grande coleção de dados que, em princípio, pode prover visões dos dados que não são possíveis em códigos transacionais individuais (BRADLEY *et al.*, 1998).

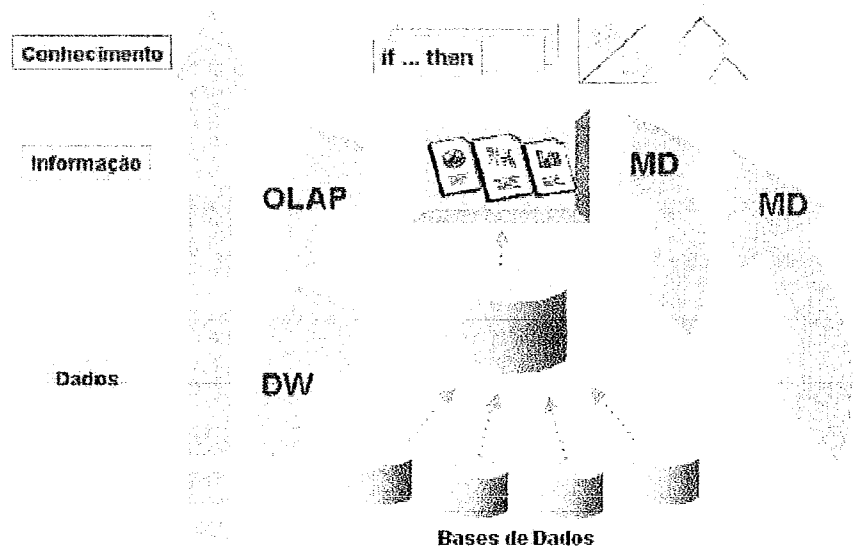


Figura 2.1 - Processo de Extração do conhecimento (REZENDE *et al.*, 2003)

Segundo Inmon (INMON, 1996) para o processo de MD não é necessário ser implementado um *Data Warehouse*. Entretanto, caso se tenha um DW já implementado, o tempo gasto na etapa de pré-processamento é reduzido drasticamente. Com isso, os analistas podem concentrar a atenção em outras tarefas importantes como a extração de padrões, avaliação e consolidação de conhecimento.

Os resultados de uma ferramenta OLAP são obtidos através de consultas apenas em cima de um banco criado a partir de um processo de DW. Enquanto isso, um processo de Mineração de Dados pode ser feito tanto diretamente em um conjunto de base de dados quanto em um banco produzido por um DW, como pode ser visto na Figura 2.1.

A Figura 2.1 mostra uma síntese do processo de extração do conhecimento, originalmente apresentada em (REZENDE *et al.*, 2003). As etapas envolvidas nesse processo são detalhadas logo a seguir.

2.1 – Etapas do processo de mineração de dados

Existem diversas abordagens para a segmentação das etapas do processo de extração de conhecimento de bases de dados. A divisão inicialmente proposta por (BRADLEY *et al.*, 1998) particiona o processo em nove etapas. Já em (WEISS *et al.*, 1998), a divisão é feita em apenas quatro etapas. Entretanto, a divisão adotada nesta

dissertação é a sugerida por (REZENDE *et al.*, 2003), que divide o processo em três grandes etapas: pré-processamento, extração de padrões e pós-processamento.

2.1.1 – Etapa de pré-processamento

Na maioria dos casos em que se busca um conhecimento a partir de uma base de dados, os dados não estão em um formato adequado à extração de conhecimento, às vezes, sequer estão concentrados em um mesmo tipo de formato. Além disso, tem-se o problema da dimensionalidade, onde a base de dados é muito grande e as restrições de memória e de processamento inviabilizam o seu tratamento direto.

A fim de transformar os dados em uma forma tratável computacionalmente é feito um pré-processamento. Vale ressaltar que a execução das transformações deve ser guiada pelos objetivos do processo de extração, a fim de garantir que o conjunto de dados gerado apresente as características necessárias para que os objetivos sejam cumpridos.

Além disso, cada algoritmo da etapa de extração de padrões pode ter requisitos diferentes de funcionamento e pode exigir um tratamento diferenciado nessa etapa.

As principais etapas deste processo são (REZENDE *et al.*, 2003): extração e integração, transformação, limpeza e seleção e redução dos dados. Essas etapas são descritas com mais detalhe a seguir.

2.1.1.1 – Extração e integração

Nessa etapa, ocorre a formação de uma única fonte de dados a partir de arquivos de texto, arquivos no formato de planilhas, banco de dados e *Data Warehouse*. Podem ser utilizadas duas técnicas nesse processo: junção direta ou junção orientada.

Na junção direta capturam-se todos os atributos de todas as tabelas que serão analisadas, colocando-os em uma tabela única.

Na junção orientada, utiliza-se a ajuda do especialista do domínio e do especialista de KDD. Os atributos são escolhidos de forma que apenas aqueles que podem contribuir para a análise são levados em consideração.

2.1.1.2 – Transformação

Alguns algoritmos de mineração de dados necessitam que os dados estejam em um certo formato para que eles possam ser executados normalmente. Por isso, algumas transformações são feitas nessa etapa.

Existem duas transformações possíveis dependendo da necessidade de cada algoritmo de mineração. Essas transformações podem ser divididas em duas tipologias básicas: *Numérica → Categórica* e *Categórica → Numérica.*

Para algoritmos que não aceitem valores numéricos como entrada é adotada a transformação *Numérica → Categórica* onde é feita a substituição de valores numéricos por valores categóricos. Pode-se fazer um mapeamento direto dos dados, por exemplo: $0 \rightarrow \mathrm{F}$, $1 \rightarrow \mathrm{M}$; ou então por intervalos, por exemplo: $[0, 5] \rightarrow \text{Ruim}$, $[6, 8] \rightarrow \text{Bom}$ e $[8, 10] \rightarrow \text{Ótimo}$.

Para algoritmos que não aceitem valores alfanuméricos como entrada é adotada a transformação *Categórica → Numérica* onde é feita a substituição de valores normalmente textuais por valores numéricos. Pode ser feita subdividida em três tipos: representação binária padrão, representação binária *1-de-N* e representação binária por temperatura.

Na representação binária padrão, analisa-se a quantidade de valores categóricos diferentes para a característica, ou atributo, e atribui-se uma codificação binária para cada valor. Por exemplo, o estado civil de um cidadão pode ser: casado, solteiro, divorciado, viúvo ou separado (cinco valores). Poderia ser representado da seguinte maneira:

Casado – 001

Solteiro – 010

Viúvo – 100

Divorciado – 011

Separado – 110

Na representação binária *1-de-N,* o tamanho da cadeia de bits para representar o valor categórico tem o tamanho da quantidade de valores possíveis para o atributo, ou seja, nesse caso cada cadeia terá cinco bits que a define. Nesse caso a transformação do exemplo anterior seria feita da seguinte forma:

Casado – 00001

Solteiro – 00010

Viúvo – 00100

Divorciado – 01000

Separado – 10000

Na representação binária por temperatura, assim como na anterior, o tamanho da cadeia de bits para representar o valor categórico tem o tamanho da quantidade de valores possíveis para o atributo. Porém ele indica além do dado categórico uma relação entre eles. Essa representação é muito utilizada quando os valores discretos estão relacionados e onde existe uma relação de ordem ou graduação entre os valores. Por exemplo:

Péssimo – 0001

Ruim – 0011

Bom – 0111

Ótimo – 1111

2.1.1.3 – Limpeza dos dados

Essa etapa tem como finalidade retirar ruídos que possam ter ocorrido nos dados presentes para análise. Nela é feita a retirada ou correção dos erros de digitação ou erros na leitura dos dados pelos sensores. Inconsistências e anomalias também são detectadas durante a execução dessa etapa. Basicamente, pode-se detalhar a etapa de *Limpeza dos dados* em três principais tipos: limpeza de informações ausentes, limpeza de inconsistências e limpeza de valores não pertencentes ao domínio da aplicação.

Na limpeza das informações ausentes é feita uma verificação nos atributos que compõem cada observação para se assegurar se nenhuma delas está com informação faltante, exceto para aqueles que por definição podem conter informações ausentes. Quando uma ausência é detectada, quatro ações podem ser tomadas: exclusão dos dados, preenchimento manual do dado faltante, preenchimento com valor constante e preenchimento com medidas estatísticas.

Na limpeza de inconsistências é feita uma verificação nos valores dos atributos que compõem cada observação para se assegurar que nenhuma delas viole as regras do problema. Duas ações podem ser tomadas nesse caso: exclusão dos dados ou correção manual do dado inconsistente.

Na limpeza de valores não pertencentes ao domínio é feita uma verificação nos valores dos atributos que compõem cada observação para se assegurar que nenhum

deles viole os valores permitidos no domínio da aplicação. Nesse caso as ações possíveis são semelhantes às anteriores: exclusão dos dados ou correção manual do dado fora do domínio.

2.1.1.4 – Seleção e redução dos dados

Nessa etapa é feita a redução na dimensionalidade do problema. Pode ser feita através da redução no número de exemplos, de atributos ou de valores de um determinado atributo.

A redução no número de exemplos deve ser feita de tal forma que o conjunto de dados gerado possua as mesmas características do conjunto original, ou seja, preserve as mesmas características do conjunto base. Essa redução pode ser feita de duas formas principais: por segmentação do banco de dados ou por amostragem.

Na segmentação do banco escolhe-se um ou mais atributos que servirão de base para a escolha dos registros. Por exemplo, em uma tabela de Países, serão avaliados apenas os países pertencentes ao continente africano. Um simples comando SQL pode resolver o problema e mostrar apenas os dados selecionados:

```
SELECT p.COD, p.NOME, p.CONTINENTE, p.POPULACAO
FROM TAB_PAIS p
WHERE p.CONTINENTE = 'AFRICA'
```

A redução por amostragem pode ser feita com reposição, sem reposição ou de forma estratificada. Na redução sem reposição cada observação tem a mesma probabilidade de ser sorteada, sendo a probabilidade de cada uma igual a n / N, onde n é o tamanho da amostra e N é a quantidade total de observações disponíveis. Cada observação escolhida é retirada do conjunto total. Já na redução com reposição a probabilidade permanece a mesma, porém a observação escolhida permanece no conjunto total, podendo ser escolhida novamente. Na amostragem estratificada são criados grupos, denominados estratos, no conjunto de observações. Após a criação desses grupos alguns subconjuntos de cada grupo são selecionados para a amostra. Nesse tipo de amostragem há uma maior representatividade dos dados e precisão nas inferências desde que os estratos sejam definidos por variáveis que tenham correlação com o que for analisado.

A abordagem mais utilizada para redução do número de exemplos é a amostragem aleatória (WEISS *et al.*, 1998), pois este método é mais simples e não requer maiores estudos. Se a amostra não for representativa, ou se a quantidade de observações for insuficiente para caracterizar os padrões embutidos nos dados, os modelos produzidos podem não representar a realidade. Desta forma, a execução correta dessa fase mostra-se vital ao processo como um todo.

A redução no número de atributos pode ser feita tanto de forma independente do modelo quanto de forma dependente. Na abordagem independente, os atributos são eliminados mesmo sem se saber qual algoritmo será utilizado nos atributos selecionados. Já na abordagem dependente, testa-se o algoritmo para cada conjunto de atributos e verifica-se qual a melhor opção. Como pode ser facilmente imaginado, esse processo requer mais tempo e poder de processamento.

A configuração dos atributos que devem ser escolhidos pode ser feita de três maneiras distintas. Na primeira, tem-se um conjunto inicial de atributos vazio. A cada iteração um novo atributo é adicionado e avalia-se esse conjunto segundo alguma medida já existente. Já na segunda maneira, o conjunto inicial possui todos os atributos e a cada iteração um atributo é retirado desse conjunto, segundo alguma medida de avaliação. A terceira maneira é uma combinação das duas anteriores. A cada iteração o melhor atributo é incorporado ao conjunto e o pior é eliminado.

A terceira forma de redução dos dados é através da redução no número de valores de um atributo. São utilizadas, geralmente, duas maneiras para essa tarefa, a discretização ou a suavização dos valores de um atributo.

A discretização de um atributo consiste na substituição de um atributo contínuo (inteiro ou real) por um atributo discreto, por meio do agrupamento de seus valores (REZENDE *et al.*, 2003). Os métodos de discretização podem ser classificados em supervisionados ou não-supervisionados, locais ou globais, e parametrizados ou não-parametrizados (FÉLIX *et al.*, 2000).

A suavização dos valores de um atributo tem como objetivo diminuir o número de valores do mesmo sem discretizá-los. Nesse método, os valores de um determinado atributo são agrupados, mas, ao contrário da discretização, cada grupo de valores é substituído por um valor numérico que o represente (WEISS *et al.*, 1998).

Os métodos e algoritmos utilizados nessa etapa de pré-processamento são executados antes da extração de padrões. Entretanto, como Mineração de Dados é um processo iterativo, algumas dessas atividades podem ser realizadas novamente após a

análise dos padrões encontrados na etapa de extração de padrões. Por exemplo, acrescentar um determinado atributo, reduzir ou aumentar o conjunto de observações, para que o indutor possa utilizá-lo de forma mais eficiente, e melhorar a qualidade do conhecimento extraído.

2.1.2 – Extração de padrões

Após o pré-processamento, os dados já se encontram de uma forma adequada à aplicação de um algoritmo de extração de padrões. Nessa etapa, é realizada a escolha, a configuração e a execução de um ou mais algoritmos de extração de conhecimento. O algoritmo escolhido nessa etapa é, em geral, executado diversas vezes visando um melhor ajuste dos parâmetros ou a melhoria da compreensibilidade do conhecimento extraído.

As técnicas de Aprendizado de Máquina parecem estimular muito os pesquisadores da área e talvez seja a parte de processo de Mineração de Dados que contenha substancialmente os maiores conteúdos científicos estudados (REZENDE *et al.*, 2003).

Essa etapa pode ser subdividida em três fases: escolha da tarefa de Mineração de Dados, escolha do algoritmo e extração dos padrões.

2.1.2.1 – Escolha da tarefa

A escolha da tarefa de Mineração de Dados é feita de acordo com os objetivos a serem atingidos com a solução do problema. Existem dois tipos básicos de tarefas, as preditivas e as descritivas. A Figura 2.2 mostra uma árvore classificatória das diversas opções de Mineração de Dados, em diversos níveis.

As atividades preditivas, dentro do enfoque de Mineração de Dados preditivo, consistem em predizer um resultado futuro com base nos resultados atuais ou em experiências passadas com respostas conhecidas. Essas atividades visam principalmente à tomada de decisões. Na realização dessa atividade podem ser utilizados dois tipos de algoritmos, de classificação e de regressão. O uso de cada um deles depende do problema a ser estudado.

Algoritmos de classificação são utilizados em dados discretos, ou categóricos. Por exemplo, em uma análise cancerígena predizer se o tumor de um determinado paciente é maligno ou benigno. Já os algoritmos de regressão trabalham em cima de

dados contínuos, por exemplo, predizer o lucro ou a perda em um empréstimo (WEISS *et al.*, 1998)

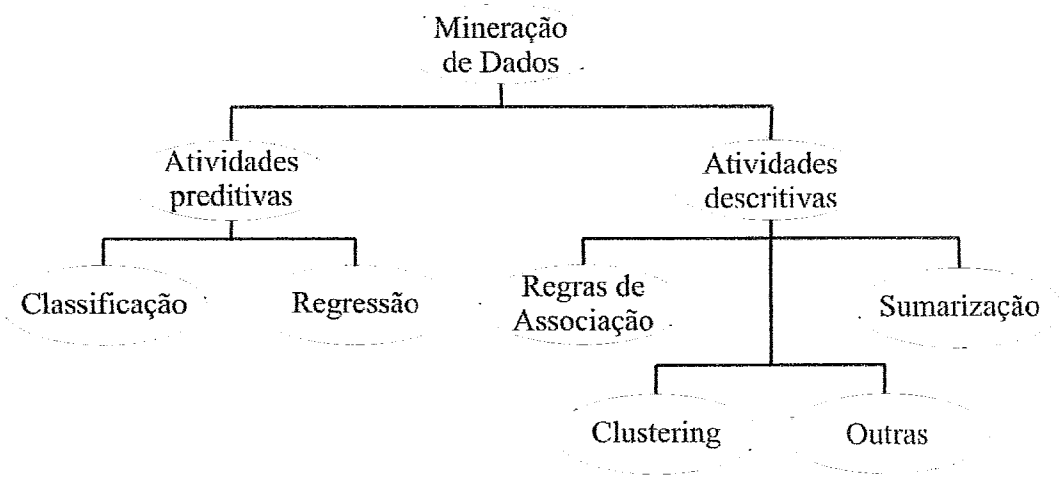


Figura 2.2 - Tarefas da Mineração de Dados (REZENDE *et al.*, 2003)

As atividades descritivas, dentro do enfoque de Mineração de Dados descritivo, consistem em identificar comportamentos intrínsecos ao conjunto de dados. Entretanto, não se tem uma informação classificatória a priori. Atividades desse tipo visam o suporte à decisão. Procuram padrões facilmente interpretáveis pelos humanos que descrevem os dados. Dessa forma, o padrão encontrado pode ser utilizado pelo gestor na tomada de decisão. Os principais algoritmos aplicados nesta atividade são: agrupamento ou *clustering*, regras de associação e sumarização.

Após a escolha da atividade a ser realizada, é feita escolha do algoritmo a ser empregado e a configuração dos parâmetros para a execução do mesmo.

2.1.2.2 – Escolha do algoritmo

A escolha do algoritmo é feita seguindo as definições estabelecidas no passo anterior. De acordo com o tipo de atividade é escolhido um método de solução do problema. Se a atividade escolhida for preditiva e uma tarefa de classificação, por exemplo, deve-se escolher o critério e, então, definir agora qual o algoritmo de classificação a ser utilizado. Os principais algoritmos de classificação são: árvores de decisão, modelos lineares, modelos não-lineares fundamentadas em Redes Neurais Artificiais, modelos baseados em exemplos, como *K-Nearest Neighbor* (KNN) e

Support Vector Machines (SVM), e modelos de dependência probabilística conhecidas como Redes Bayesianas. Já para atividades descritivas, cuja tarefa seja o agrupamento, deve-se optar por algoritmos como: K-Means, agrupamento hierárquico ou de particionamento. Além dos algoritmos citados, as Cadeias de Markov são amplamente aplicadas na descoberta de padrões.

A sugestão dada por (KEARNS *et al.*, 1994) para escolher uma determinada função é: o modelo mais apropriado é aquele mais simples que seja consistente com todas as observações. Normalmente, soluções mais complexas são preferidas por pesquisadores, enquanto os práticos tendem a preferir modelos mais simples em virtude de sua fácil interpretação (BRADLEY *et al.*, 1998). A escolha de mais de um algoritmo também pode ser feita. Nessa hipótese, na etapa de pós-processamento as diversas soluções são analisadas e a solução mais adequada é escolhida.

2.1.2.3 – Extração de padrões

É nessa etapa que o algoritmo de mineração escolhido é realmente aplicado aos dados para extração dos padrões desejados. Normalmente, um mesmo algoritmo é aplicado várias vezes para que os parâmetros possam ser ajustados da melhor maneira possível.

Algumas pesquisas têm sido desenvolvidas no sentido de se criar um preditor a partir da combinação de vários outros, de forma a obter melhores resultados. Segundo (DIETTERICH, 2000), essa combinação é denominada *ensemble* e, em geral, tem obtido melhores resultados que a utilização de um único preditor.

2.1.3 – Pós-processamento

Nessa etapa, o conhecimento já foi extraído e está pronto para ser utilizado, seja por meio de um sistema inteligente ou como apoio a algum processo de tomada de decisão. Nesse momento, três questões devem ser observadas de forma a fazer uma validação dos dados obtidos (LIU *et al.*, 1996):

- O conhecimento extraído representa o conhecimento do especialista?
- De que maneira o conhecimento do especialista difere do conhecimento extraído?
- Em que parte o conhecimento do especialista está correto?

Diversas medidas para avaliação de conhecimento têm sido estudadas com a finalidade de auxiliar o usuário no entendimento e na utilização do conhecimento obtido. Medidas de desempenho como: precisão, erro, confiança negativa, especificidade, cobertura, suporte, satisfação, velocidade e tempo de aprendizado (LAVRAC *et al.*, 1999) são amplamente utilizadas nesse contexto.

Um dos fatores mais importante no processo de extração do conhecimento é a compreensibilidade do conhecimento extraído. É esperado que o usuário possa compreender e utilizar o conhecimento descoberto. Em geral, os usuários especialistas possuem uma tendência a compreender melhor modelos que apresentem um menor número de regras e que não contradizem o seu conhecimento prévio.

Após a avaliação do conhecimento, caso esse não seja de interesse do usuário final ou não cumpra com os objetivos propostos, o processo de extração pode ser repetido, ajustando-se os parâmetros ou melhorando o processo de escolha dos dados para a obtenção de resultados m ais adequados numa próxima iteração.

Capítulo 3 – Classificação

A classificação de dados é um problema amplamente estudado nas áreas de aprendizado de máquinas, estatística, banco de dados, otimização, computação paralela e de alta performance. Como o próprio nome diz, busca-se distinguir os dados de acordo com um determinado critério. Por exemplo, verificar se um determinado tipo de cogumelo é comestível ou não, se um tumor é maligno ou benigno, se um cliente é bom ou mal pagador, enfim, distinguir dados segundo categorias.

Um outro problema correlato é o agrupamento, onde determinados elementos são agrupados de acordo com semelhanças e diferenças. Por exemplo, em um estudo étnico agrupa-se os indivíduos nas diversas etnias existentes.

Basicamente, essas duas áreas de pesquisa se distinguem quanto ao tipo de separação. No processo de classificação, tem-se um número conhecido de conjuntos de dados e procura-se rotular determinado elemento de acordo com o seu grau de pertinência a um determinado conjunto. Já o processo de agrupamento de dados é um pouco mais primitivo, não se sabe sequer o número de grupos existentes no conjunto total de dados ou observações. Busca-se um arranjo natural dos dados de acordo com o seu grau de similaridade com os elementos do mesmo grupo e dissimilaridade com os elementos de grupos diferentes. No entanto, os dois problemas estão intrinsecamente relacionados, pois podemos encontrar problemas de agrupamento com um número determinado de conjuntos. Desta forma, um mesmo problema pode ter um enfoque de tratamento segundo uma visão de classificação ou de agrupamento.

O problema clássico de classificação pode ser definido matematicamente como (BRADLEY *et al.*, 1998): dado um conjunto S com l observações, $S=\{s_1,...,s_l\}$, atribuir a um dado vetor $s \in S$ uma das k classes disjuntas $C_1, C_2, ..., C_k$ de S. Uma simplificação do problema pode ser feita e é amplamente utilizada: determinar a qual classe A ou B pertence um determinado vetor s, sendo $A, B \subset S$. Dessa forma pode-se assumir uma função de classificação f da seguinte maneira:

$$f(s)=\begin{cases}1 & se\ s \in A \\ 0 & se\ s \in B\end{cases} \tag{3.1}$$

Existem vários métodos que procuram aproximar uma função $\hat{f}$ da função f. Essa aproximação pode ser feita de diferentes formas. Por exemplo, as funções baseadas em um plano separador (MANGASARIAN, 1965; VAPNIK, 1995), o algoritmo de retropropagação (*backpropagation*) para redes neurais artificiais (MCCLELLAND *et al.*, 1987; J. HERTZ *et al.*, 1991), algoritmos de construção de árvores de decisão utilizando vários nós de critério de decisão (BREIMAN *et al.*, 1983; BENNETT, 1992; QUINLAN, 1993), métodos "*spline*" para classificação (WAHBA *et al.*, ; WAHBA, 1990) e modelos gráficos de dependência probabilística (BUNTINE, 1996; HECKERMAN, 1997).

No processo de classificação tem-se, em geral, um conjunto de dados de treinamento e outro de teste. As observações do conjunto de treinamento contêm dois pares de dados, um vetor $x \in \Re^n$ de características e um rótulo y que indica a qual classe determinada observação pertence. Já o conjunto de teste possui apenas o conjunto de características. Os métodos de classificação procuram inferir essa relação $x_i \mapsto y_i$, $i = 1,...,l$. Para isso, "treina-se" uma determinada função $f(x,\alpha)$ onde o parâmetro α é ajustado de forma a produzir o resultado esperado para y. Esta função deve ser determinística, ou seja, para um dado x e um valor para o parâmetro de treinamento α, deve-se obter sempre o mesmo valor em $f(x,\alpha)$.

Existem dois tipos de erros intrínsecos ao processo de classificação, o erro de treinamento, ou risco empírico, e o erro de classificação. O erro de treinamento está ligado ao processo de aprendizagem do classificador, ou seja, ao processo de "calibragem" do parâmetro α. Esse erro é calculado levando-se em consideração apenas o conjunto de treinamento e é dado pela seguinte equação:

$$R_{emp}(\alpha) = \frac{1}{2l}\sum_{i=1}^{l}\left|y_i - f(x_i,\alpha)\right|. \tag{3.2}$$

sendo:

- R_{emp} o erro, ou risco, empírico
- α um dado vetor de parâmetros da função de classificação
- l o número de observações
- i uma observação genérica

- y_i o rótulo da observação i
- f a função de classificação adotada
- x_i o conjunto de características da observação i.

A parcela $\frac{1}{2}|y_i - f(x_i, \alpha)|$, associada a cada observação i, representa a perda no processo de aprendizado e pode ter valores 0 e 1, dado que o rótulo das observações sejam iguais a 1 e -1. O termo correspondente ao somatório é igual ao número de erros cometidos no processo de treinamento.

A equação (3.2) mostra que o erro de treinamento é um valor determinístico e invariável para um dado conjunto de treinamento e um conjunto de parâmetros α, pois depende apenas da diferença entre o rótulo e o valor dado pela função $f(x,\alpha)$. O mesmo não acontece com o erro de classificação, pois o cálculo do mesmo depende da função densidade de probabilidade dos dados $p(x,y)$. Assim, fazendo-se a integração no domínio das variáveis aleatórias (x,y), tem-se a seguinte equação para o erro de classificação:

$$R(\alpha) = \int \frac{1}{2}|y - f(x,\alpha)|p(x,y)\,dxdy \tag{3.3}$$

As equações acima mostram que o cálculo do erro de classificação, dado por (3.3) , não pode ser feito precisamente, ao contrário do erro de treinamento, dado por (3.2), que é exato para dado conjunto $\{x_i, y_i\}$. Segundo (BURGES, 1998), escolhendo-se η tal que $0 \leq \eta \leq 1$ pode-se obter o seguinte limite para o erro de classificação com probabilidade de $1-\eta$:

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\left(\frac{h\left(\log(2l/h)+1\right) - \log(\eta/4)}{l}\right)} \tag{3.4}$$

onde h é um inteiro não-negativo chamado dimensão VC (Vapnik Chervonenkis), que será apresentado na próxima seção.

o valor de $p(x,y)$, assume-se apenas que tanto o conjunto de treinamento como o conjunto de teste possuem uma certa mesma densidade de probabilidade $p(x,y)$. Segundo, não é possível se calcular com precisão o lado esquerdo da equação (3.4). Enfim, conhecendo o valor da dimensão VC, é possível calcular com facilidade o lado direito da equação. Assim, pode-se obter uma máquina classificadora com o menor limite superior para o risco esperado.

O poder de classificação de um classificador, ou seja a qualidade do classificador, é medido pela capacidade de generalização do modelo. Na fase de treinamento, procura-se qualidade do conjunto de treinamento: representatividade do universo e precisão das medidas. Adicionalmente, o aspecto quantidade deve ser também considerado: um número reduzido de observações, mas suficiente para oferecer o nível de precisão desejado.

Porém, nem sempre um bom classificador no conjunto de treinamento é um bom classificador do conjunto de teste. Existem duas dificuldades comuns em classificadores: a especialização do modelo, que ocorre quando o modelo se especializa demais no conjunto de treinamento, e a utilização de mais características que as necessárias.

Basicamente, há três problemas oriundos da redução na capacidade de generalização de um classificador (JAIN *et al.*, 2000) :

- sobre-ajuste (*overfitting*), dependência excessiva do conjunto de treinamento, relacionado com o número de parâmetros livres do classificador;
- sobre-treinamento (*overtraining*), relacionado com o número de iterações na fase de treinamento do modelo;
- problema da dimensionalidade (*curse of dimensionality*), relacionado com a dimensão do espaço de características.

Desta maneira, a qualidade do classificador pode ser medida por uma relação entre a qualidade do conjunto de treinamento (representatividade dos dados), o número de características dos dados e a complexidade do classificador.

Como pode ser visto na Figura 3.1, há uma forte relação entre o erro de treinamento e o erro de teste. Deve-se tomar cuidado para que o modelo não se ajuste excessivamente às observações do conjunto específico de treinamento, fugindo a

representatividade geral do universo dos dados. A busca pelo equilíbrio entre o *overfitting* e o *underfitting* é uma tarefa difícil e nem sempre atingida.

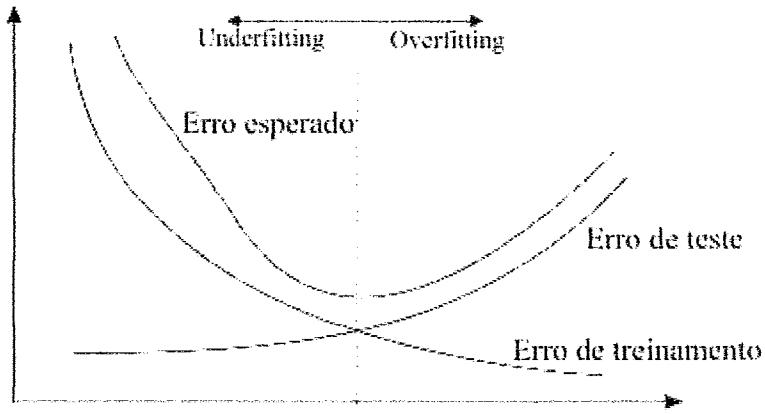


Figura 3.1 - Relação entre os erros de classificação

Uma analogia ao problema pode ser feita através do dilema do alfaiate. O que é melhor, uma roupa sob medida ou *prêt-à-porter*? Não há uma resposta exata para esta pergunta. Fábricas de roupas procuram um ajuste nas dimensões de suas roupas de forma que as peças sirvam a um maior número de consumidores (generalização). E alfaiates e costureiras buscam o melhor ajuste para um determinado cliente. De acordo com o alvo a ser atingido optamos por um *prêt-à-porter*. Os problemas de classificação buscam um padrão análogo às fábricas de roupas, onde a generalização é o alvo principal.

Um dos métodos utilizados para se testar essa capacidade é o *leave-one-out* (LAHENBRUCH *et al.*, 1968). O método consiste em se executar o algoritmo de classificação n vezes e a cada execução é deixado um vetor do conjunto de treinamento de fora do processo para ser utilizado como teste. Com isso é possível se calcular uma média do erro obtido durante todo o processo. Uma outra forma de se estimar o erro de generalização do modelo é através do *Cross-Validation* (STONE, 1974). Neste processo, o conjunto de treinamento é dividido em l conjuntos disjuntos de tamanhos aproximadamente iguais $T_1, T_2, ..., T_l$. O algoritmo é executado l vezes, cada uma delas deixando o conjunto T_i como conjunto de teste e fazendo o conjunto de treinamento como a união dos conjuntos restantes, como na equação abaixo.

$$T = \bigcup_{j=1,\, j\neq i}^{l} T_j \tag{3.5}$$

Como pode ser observado, o método de avaliação *cross-validation* apresenta um ganho em tempo de execução com relação ao *leave-one-out*, o que, em problemas com um elevado número de elementos, pode ser bastante significativo. Entretanto, não apresenta nenhuma inovação, visto que é apenas uma especificação (simplificação) do algoritmo original de *leave-one-out*.

3.1 – Dimensão VC

A dimensão de Vapnik Chervonenkis (VC) é uma propriedade de um conjunto de funções $\{f(\alpha)\}$ e pode ser definida através da várias classes de funções f. Sem perda de generalidade, considera-se as funções de separação em duas classes, $f(x,\alpha) \in \{-1, 1\}\ \forall x,\alpha$. Se um conjunto de pontos pode ser separado em 2^l formas distintas, sendo l o número de características de cada observação do conjunto e para cada separação um membro do conjunto $\{f(\alpha)\}$ pode ser obtido de forma que cada elemento tenha sido separado de forma correta, diz-se que o conjunto pode ser separado pelo conjunto de funções (BURGES, 1998). A dimensão VC de um conjunto de funções $\{f(\alpha)\}$ é definida como o número máximo de pontos de treinamento que podem ser separados por esse conjunto $\{f(\alpha)\}$. Assim, se a dimensão VC é h, então existe pelo menos um conjunto de h pontos que podem ser separados. O inverso dessa relação nem sempre pode ser satisfeito, visto que nem todo conjunto de h pontos pode ser separado pelo mesmo conjunto de funções.

Considere o conjunto de hiperplanos no $\Re^n$. A dimensão VC deste conjunto é $n+1$, pois existem n vetores linearmente independentes no espaço $\Re^n$. Assim, o maior número de elementos que podem ser linearmente separados neste espaço é $n+1$. No espaço $\Re^2$, por exemplo, o conjunto de pontos deve conter apenas 3 elementos.

A Figura 3.2 mostra que para um conjunto de três elementos no $\Re^2$ a separabilidade linear é sempre garantida.

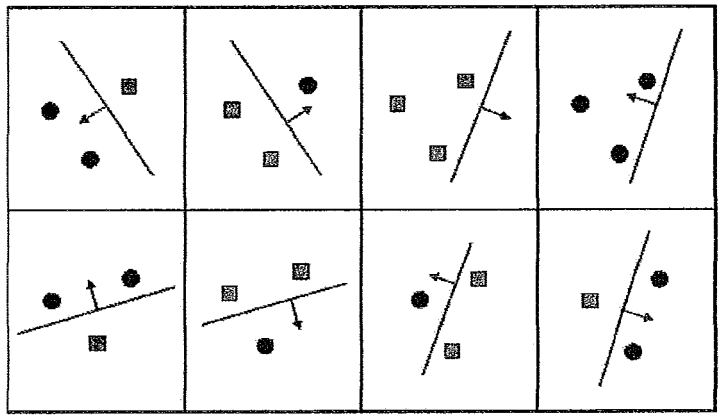


Figura 3.2 - Dimensão VC com 3 pontos.

A Figura 3.3 mostra que para um conjunto com quatro elementos a separabilidade linear já não pode mais ser garantida em todos os casos. Um exemplo clássico é o problema "ou exclusivo" (XOR, *exclusive or*), representado nos dois últimos quadros da figura abaixo.

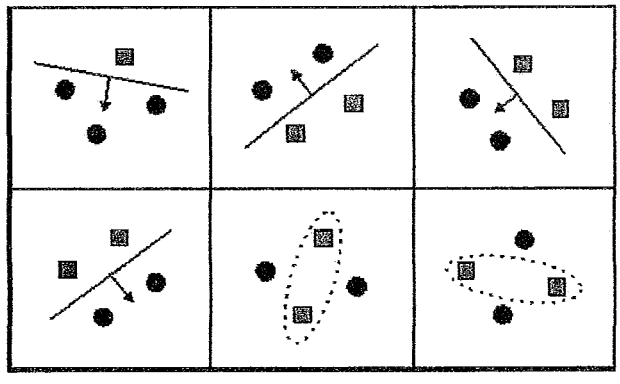


Figura 3.3 - Não separabilidade linear com 4 pontos.

3.2 – Métodos de Classificação

Como dito anteriormente, no processo de classificação o objetivo é encontrar o rótulo, ou classe, de um determinado elemento a partir de um conjunto de características dadas. Como não se tem um conhecimento a priori da densidade dos conjuntos e essa densidade é difícil de se calcular, foram criadas várias técnicas para estimar a mesma, possibilitando assim, a classificação dos dados. Essas técnicas estão divididas segundo (BRADLEY *et al.*, 1998) em:

Estimativa de densidade: por exemplo, estimativas de densidade Kernel (R. O. DUDA, 1973) e representações gráficas de densidade (HECKERMAN, 1997).

Métodos baseados em um espaço métrico: define-se uma medida de distância entre os pontos e supõe-se o valor da classe de acordo com a proximidade dos pontos do conjunto de teste com os pontos do conjunto de treinamento. Por exemplo, o método que utiliza a técnica do k - vizinho mais próximo definido por Duda e Hart (R. O. DUDA, 1973).

Projeções em regiões de decisão: divide-se o espaço de entrada (atributos) em regiões de decisão e associa-se um rótulo a cada região. Por exemplo, a análise de discriminante linear determina separadores lineares e as redes neurais produzem superfícies de decisão.

A seguir, são descritos alguns exemplos de métodos de classificação.

3.2.1 – Método dos k - vizinhos mais próximos

O método dos k - vizinhos mais próximos (*K-nearest neighbor KNN*) diferencia-se dos outros por não possuir processamento na fase de treinamento, ou seja, não é necessário gerar um hiperplano classificador e não se estima uma probabilidade de distribuição das classes. As fases de treinamento e de teste são feitas concomitantemente durante o mesmo processo e é necessário um grande número de observações de treinamento, cuja classe é conhecida a priori.

O algoritmo KNN é feito em três etapas:

- Dado uma nova observação de teste s, calcula-se a distância entre s e todas as observações de treinamento;
- Determina-se o conjunto das classes associadas às k observações mais próximas da observação s;
- A nova observação é classificada de acordo com a classe mais freqüente no conjunto acima.

As duas distâncias geralmente adotadas na implementação deste método são as distâncias Mahalanobiana e Euclidiana, descritas no Apêndice 1.

A Figura 3.4 ilustra o comportamento do algoritmo com $k = 5$. O primeiro quadro mostra todas as distâncias ao novo ponto s. O segundo quadro as cinco menores distâncias. Nesse exemplo, o ponto s é classificado como pertencente à classe

representada por círculos, pois, como pode ser observado, existem três menores distâncias da classe de círculos e apenas duas da classe de quadrados.

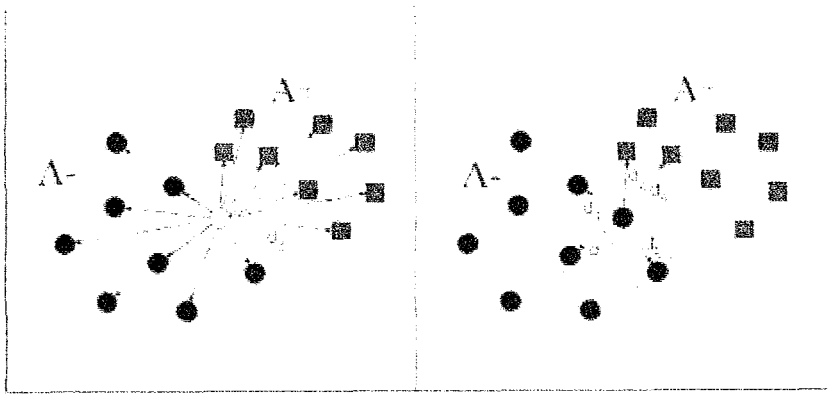


Figura 3.4 - Algorítmo KNN com $k = 5$

A principal vantagem do método é a criação dinâmica de uma superfície de decisão que molda de forma adequada a distribuição dos dados de treinamento, fazendo com que o método apresente boas taxas de acerto em problemas cujo conjunto de treinamento T é grande ou representativo.

Como pode ser facilmente observado, tomando-se $k=1$ no método dos k - vizinhos mais próximos tem-se o método do vizinho mais próximo (*Nearest Neighbor NN*) utilizado com freqüência em aplicações de reconhecimento de faces.

O objetivo de se utilizar $k>1$ é reduzir os erros causados por ruídos nas observações de treinamento. Por exemplo, se uma observação de treinamento s_i da classe 1 se encontrar em uma região do espaço de características que contém observações da classe 2 devido à ação de ruídos, essa ocorrência não prejudicará a eficiência do classificador (Figura 3.5). Pois a classificação de uma nova observação de teste será definida pela classe dominante no seu entorno e não apenas pelo ponto mais próximo. Assim, uma nova observação que se localiza próximo à observação s_i será classificada como pertencente a classe 2. Desta forma, o uso do método KNN é mais indicado em relação ao NN para conjuntos com observações com classificação incorreta, devido a ruído, e quando se tem um conjunto de treinamento com muitos exemplos. Entretanto, quando a distribuição das classes possui muitas sobreposições, o uso de valores grandes para k pode reduzir a qualidade dos resultados da classificação pela confusão estabelecida por essa configuração.

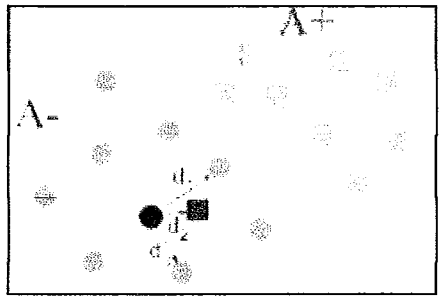


Figura 3.5 - Exemplo de classificação com ruído e $k = 3$

Como pode ser observado, além da escolha do tipo de métrica a ser utilizada, um outro fator importante para o bom desempenho do método é a escolha do valor de k. Não há uma regra definitiva para essa escolha, o que é utilizado normalmente é o método da tentativa e erro. São testados alguns valores para k e é adotado o valor que produza melhores resultados de classificação. Theodoridis e Koutroumbas (THEODORIDIS *et al.*, 2006) sugerem que para $k \to \infty$ quando $|T| \to \infty$, o desempenho do classificador KNN tende a ser ótimo. Entretanto resultados práticos vêm mostrando que o uso de $k = 3$ para conjuntos de treinamento numerosos traz bons resultados, chegando a se aproximar do desempenho de um classificador Bayesiano. Vale ressaltar que o valor de k deve ser sempre superior ao número de classes para que não ocorra empate na determinação da classe.

Um ponto negativo na utilização deste método é a complexidade computacional. Como pode ser observado, no pior caso ("força bruta"), a primeira etapa do algoritmo possui ordem de complexidade $k.O(|T|)$, pois quando não se tem uma ordenação prévia do conjunto de treinamento torna-se necessário calcular k vezes a distância entre todos os elementos de T. Entretanto, na implementação do método podem ser utilizadas estruturas que apóiem no armazenamento das distâncias computadas, reduzindo a complexidade para $O(|T|)$.

3.2.2 – Análise discriminante

É uma técnica de processamento estatístico que permite a classificação de dados entre duas, ou mais, classes distintas. A essência desta técnica está em se encontrar uma combinação linear de variáveis que possa distinguir da melhor forma possível os grupos

que estão sendo estudados. Tem-se uma função discriminante para cada classe e uma nova observação é dita pertencente à determinada classe de acordo com o valor encontrado na função correspondente àquela classe. Por exemplo, a observação s_1 possui valor $f_1 = 30$ para a classe 1 e $f_2 = 35$ para a classe 2. Logo, a observação s_1 é classificada como pertencente à classe 2, pois seu valor associado à função f_2 é superior ao valor associado à função f_1.

Desta forma a tomada de decisão de uma análise discriminante é feita da seguinte forma:

$$s \in \begin{cases} 1, \ se\ f_1(s) > f_2(s) \\ 2, \ caso\ contrário. \end{cases} \tag{3.6}$$

As funções discriminante lineares para $s \in \Re^n$ possuem a seguinte forma:

$$f(s_j) = w_1 s_{j_1} + w_2 s_{j_2} + \ldots + w_n s_{j_n} \tag{3.7}$$

onde:

- s_j é uma dada observação j do conjunto;
- s_{j_i} é uma componente da observação s_j;
- $f(s_j)$ é o valor da função discriminante para dada observação s_j;
- w é um vetor de pesos associado à cada observação;
- n é o número de componentes de cada observação do conjunto.

Os valores de w podem ser obtidos através de técnicas de regressão ou estimativas de probabilidade (THEODORIDIS *et al.*, 2006).

Este tipo de análise é mais utilizado em análises de crédito, para distinguir bons clientes de clientes com histórico de inadimplência, e na área de *marketing*, para distinguir clientes em potencial de determinado produto de clientes não potenciais.

Os principais pontos positivos dessa técnica são:

- permite-se trabalhar com múltiplos grupos,
- a saída pode ser modelada de várias formas. Por exemplo, em um *ranking fuzzy* de pertinência aos grupos.

Um ponto negativo a ser destacado é quando as algumas ou todas as variáveis independentes estão intimamente correlacionadas (multicolinearidade). Isso faz com que o procedimento selecione um número não razoável, dada a redundância, de variáveis na formação da função discriminante.

3.2.3 – Agrupamento (Cluster)

A análise de agrupamento é uma técnica natural e simples, na maioria das vezes, para os seres humanos. Uma criança aprende, logo cedo, a categorizar objetos e animais. Entretanto, em um ambiente computacional esse problema se torna um pouco mais complexo, pois além dos fatos e regras que distinguem os objetos, os dados podem não se apresentar de forma clara e simples.

O princípio básico de uma análise de agrupamento é a criação de grupos de observações de acordo com o seu grau de similaridade. Busca-se maximizar a semelhança entre os elementos da mesma classe e minimizar a semelhança entre os elementos de classes distintas.

Uma das áreas de aplicação da análise de agrupamento é o setor de *marketing*, onde se procura, constantemente, agrupar os consumidores em grupos de potenciais clientes de determinado produto. Desta maneira, uma propaganda de um novo produto pode ser direcionada de forma correta ao seu público alvo.

Existem duas tipologias básicas de algoritmos para se resolver problemas de agrupamentos: os hierárquicos e os não-hierárquicos ou particionais. Estes dois tipos são descritos nas seções seguintes.

3.2.3.1 – Algoritmos hierárquicos

Os algoritmos do tipo hierárquico, como o próprio nome diz, formam uma hierarquia na composição dos grupos, que pode ser representada adequadamente por uma "árvore". A definição no número de grupos é feita de acordo com a análise de qual arranjo se aproxima melhor do arranjo natural dos dados. Dessa forma, é possível definir , na árvore, se determinada junção, ou separação, deve ocorrer ou não.

Esses algoritmos podem ser subdivididos em dois subtipos: métodos de aglomeração e métodos de divisão.

Os métodos de aglomeração partem do princípio que cada observação é um conjunto, por si só. A cada iteração dois conjuntos se unem de acordo com a similaridade ou com a distância entre os mesmos, dando origem a um novo conjunto. Desta forma, no final do processo, tem-se um único conjunto constituído por todas as observações do grupo inicial. Desde que os grupos em um estágio qualquer são obtidos pela fusão de dois grupos do estágio anterior, esses métodos conduzem a uma estrutura hierárquica para os objetos (EVERITT *et al.*, 1983).

Um exemplo do funcionamento de um algoritmo do tipo hierárquico de aglomeração pode ser visualizado na Figura 3.6, através da seqüência que segue do quadro 1 ao quadro 8.

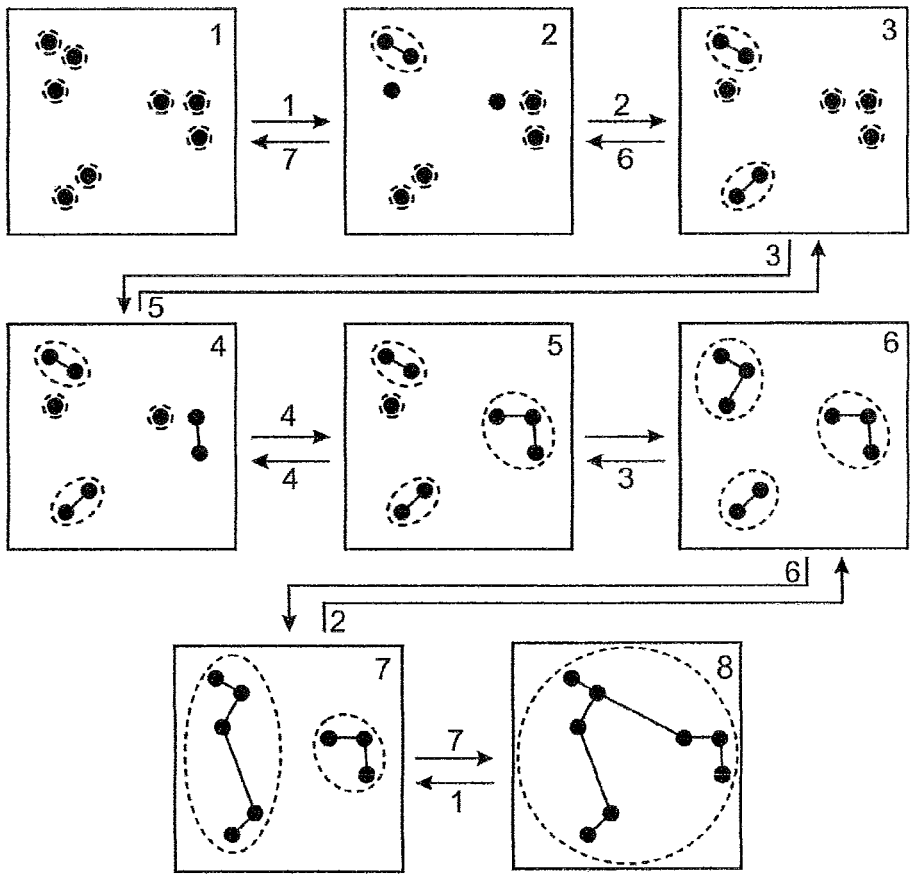


Figura 3.6 - Exemplo de agrupamento hierárquico

Os métodos de divisão são utilizados com uma freqüência menor do que os métodos por aglomeração (HANSEN *et al.*, 1997). Apresentam um caminho inverso ao descrito anteriormente. No estágio inicial todas as observações compõem um mesmo grupo. A cada iteração um grupo é subdividido em dois, de acordo com a

dissimilaridade entre as observações. A Figura 3.6, tomando-se como estágio inicial o quadro 8 e estágio final o quadro 1, ilustra o comportamento de um algoritmo hierárquico de particionamento.

3.2.3.2 – Algoritmos de partição

Os algoritmos de partição possuem uma diferença essencial dos algoritmos hierárquicos. Nesse tipo de abordagem é necessário se definir *a priori* o número de grupos. Dessa forma, o número de grupos é fixo.

Como visto anteriormente, na metodologia de algoritmos hierárquicos é possível avaliar, em cada iteração, os grupos que se formaram, permitindo a interrupção do processo iterativo quando uma conformação ideal é obtida. Por outro lado, nos algoritmos de partição, essa análise deve ser feita previamente, ou seja, não é possível verificar a cada iteração se o aumento ou a diminuição no número de grupos é vantajoso ou não.

Essencialmente, dado um número q de conjuntos e um conjunto de l observações, o algoritmo particiona essas l observações em q conjuntos. Através de análises de similaridade, dissimilaridade ou distância entre os elementos dos conjuntos, os mesmos são rearranjados de forma a melhorar a qualidade dos grupos. Em essência, busca-se simultaneamente minimizar uma função de homogeneidade intra-conjuntos e, ao mesmo tempo, maximizar uma outra função de separação entre os conjuntos.

Os conjuntos formados por essa partição em q grupos satisfazem as seguintes condições:

- Cada grupo deve conter no mínimo um objeto;
- Cada objeto deve pertencer exatamente a um conjunto.

Essas condições indicam que o número de conjuntos é, no máximo, igual ao número de elementos $q \leq l$. A segunda condição mostra que dois grupos distintos não podem possuir objetos em comum e que cada objeto pertence a um dos grupos q. A Figura 3.7 mostra a partição de 20 objetos em 3 grupos (KAUFMAN *et al.*, 1990).

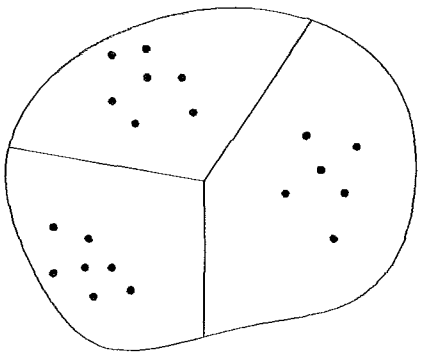


Figura 3.7 - Exemplo de agrupamento por partição

3.2.4 – Classificação via Programação Linear

O método de classificação via programação linear busca estimar uma função que classifique um dado $x \in \Re^n$ em uma das duas classes disjuntas A ou B no espaço de características n-dimensional. Tem-se $X = \Re^n, Y = \{0,1\}$ e uma função de classificação pode ser dada da seguinte forma:

$$g(x) = \begin{cases} 1 & se\ x \in A \\ 0 & se\ x \in B \end{cases} \tag{3.8}$$

A representação dos m elementos do conjunto de observações é feita pela matriz $A \in \Re^{m \times n}$ onde cada observação de A é dada por uma linha em A. Da mesma forma, os elementos do conjunto B são representados por uma matriz $B \in \Re^{\kappa \times n}$.

A separação dos pontos dos conjuntos A e B é feita através do plano separador:

$$P = \{x \mid x \in \Re^n,\ x^T w = \gamma\} \tag{3.9}$$

com normal $w \in \Re^n$ e distância $\frac{|\gamma|}{\|w\|_2}$ até a origem. O objetivo do problema é encontrar um determinado vetor w e um escalar γ, tal que o plano separador

delimite duas regiões distintas: $\{x \mid x \in \Re^n, x^T w > \gamma\}$ para os pontos do conjunto A, e $\{x \mid x \in \Re^n, x^T w < \gamma\}$ para os pontos do conjunto B. Desta forma, deve-se satisfazer as seguintes restrições:

$$\mathrm{A}w > e\gamma, \quad \mathrm{B}w < e\gamma \tag{3.10}$$

Normalizando-se as restrições acima, obtém-se as seguintes restrições normalizadas:

$$\mathrm{A}w \geq e\gamma + e, \quad \mathrm{B}w \leq e\gamma - e \tag{3.11}$$

As condições acima são satisfeitas se, e somente se, os conjuntos A e B forem disjuntos. O que na maioria das aplicações reais não é satisfeito. Assim, uma forma de tentar "satisfazer" as equações de (3.11) é através, por exemplo, da minimização de uma dada norma dos pontos que violam essas restrições como:

$$\min_{w,\gamma} f(w,\gamma) = \min_{w,\gamma} \frac{1}{m}\left\|(-\mathrm{A}w + e\gamma + e)_+\right\|_1 + \frac{1}{k}\left\|(\mathrm{B}w - e\gamma + e)_+\right\|_1 \tag{3.12}$$

lembrando que x_+ indica que, para dado vetor x, $x_+ = \max\{0, x_i\}$.

A utilização da norma um é justificada por duas razões. Primeiro, pela possibilidade de redução do problema (3.12) ao problema de programação linear (3.13), o que propicia a utilização de pacotes de solução com propriedades matemáticas teóricas importantes e propriedades computacionais de eficiência diferenciadas. E segundo, e devido ao fato da $\|.\|_1$ ser menos sensível ao pontos classificados de forma errada, chamados *outliers*.

A formulação (3.12) pode ser reescrita de forma equivalente ao problema de programação linear robusta (RLP), proposto por (BENNETT *et al.*, 1992), utilizado amplamente na resolução de problemas reais.

$$\min_{w,\gamma,y,z} \left\{ \frac{e^T y}{m} + \frac{e^T z}{k} \;\middle|\; -Aw + e\gamma + e \leq y, Bw - e\gamma + e \leq z, y \geq 0, z \geq 0 \right\} \tag{3.13}$$

O problema (3.13), ou equivalentemente, a formulação (3.12), definem um plano separador P que satisfaz as restrições (3.11).

3.2.5 – Máquina de vetor suporte - SVM

Como dito anteriormente, o problema de classificação é amplamente aplicado e estudado em várias áreas, tanto sob o enfoque teórico quanto prático. A problemática de classificação começou a ser abordada via a função discriminante de Fischer (FISHER, 1935) e, mais recentemente, ganhou uma nova metodologia de solução, através das máquinas de vetores suporte (SVM) (VAPNIK, 1995). O método SVM é essencialmente uma abordagem geométrica para o problema de classificação. Cada observação do conjunto de treinamento pode ser vista como um ponto no $\Re^n$ e o aprendizado consiste em separar os pontos positivos dos negativos.

Esta técnica procura gerar um hiperplano separador de máxima margem, o que, intuitivamente, faria aumentar o poder de generalização do classificador. Desta forma, ter-se-ia um hiperplano com maior capacidade de classificação correta para os casos do conjunto de teste (não rotulados). De forma geral, existem duas classes de problemas de separação: os linearmente separáveis e os não linearmente separáveis. Como o próprio nome sugere, na primeira classe estão os conjuntos que podem ser separados por uma superfície linear, já na segunda se faz necessária uma superfície não linear de separação, por exemplo, o problema XOR (Figura 3.8 - Problema XOR).

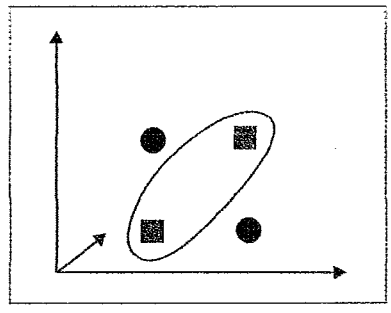


Figura 3.8 - Problema XOR

3.2.5.1 – Problemas linearmente separáveis

Para os problemas linearmente separáveis, utiliza-se uma máquina de vetor suporte linear treinada com dados linearmente separáveis, ou seja, neste caso tudo é linear. O conjunto de treinamento é composto por pontos que possuem, além das características, um rótulo da classe. Este rótulo indica a qual classe determinado ponto pertence. Assim, temos o conjunto de treinamento formado da seguinte forma: $D=\{(s^1,y^1),...,(s^l,y^l)\}, s\in\Re^n, y\in\{-1,1\}$. Onde l é a cardinalidade do conjunto de teste e n é a cardinalidade do conjunto de características.

Suponha que exista um hiperplano separador que seja capaz de separar os elementos nas classes positiva e negativa. Os pontos situados no hiperplano satisfazem a relação $x\,s+\gamma=0$, onde x é a normal ao hiperplano, $|\gamma|/\|x\|$ é a distância perpendicular do hiperplano à origem, e $\|x\|$ é a norma euclidiana de x. Tomando-se $d_+(d_-)$ como a menor distância do hiperplano positivo (negativo), tem-se a margem de separação dada por $d_+ + d_-$ (BURGES, 1998).

Para o problema de separação linear basta maximizar a margem, buscando assim, o hiperplano que possui a maior margem de separação. Na formulação a seguir supõe-se que os dados de treinamento satisfaçam as seguintes restrições:

$$s_i x+\gamma\geq+1, \qquad y_i=+1 \tag{3.14}$$

$$s_i x+\gamma\leq-1, \qquad y_i=-1 \tag{3.15}$$

As equações acima podem ser reduzidas na seguinte equação:

$$y_i(s_i x+\gamma)-1\geq 0 \qquad \forall i[0,1] \tag{3.16}$$

Considerando os pontos que satisfazem a igualdade em (3.16) tem-se os pontos situados sobre o hiperplano (pontos suporte, ou vetores suporte). Os vetores suporte podem ser definidos como pontos que ao serem removidos alteram a solução do problema (Figura 3.9).

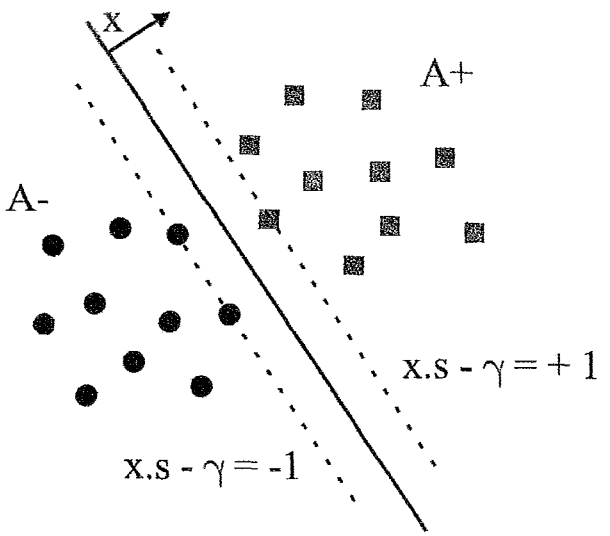


Figura 3.9 - Pontos suporte

Para a classe A+ os pontos suporte satisfazem a equação (3.14) tendo $s_i x+\gamma=1$ com normal x e com distância igual a $|1-\gamma|/\|x\|$. Da mesma forma, para a classe A- a equação (3.15) é satisfeita e tem-se $s_i x+\gamma=-1$, com normal x e com distância igual a $|-1-\gamma|/\|x\|$. Assim, a largura da margem pode ser obtida apenas por $2/\|x\|$, pois $d_+ = d_- = 1/\|x\|$.

O problema de maximização da largura da margem pode ser formulado pelo *seguinte problema de otimização quadrática:*

$$\begin{aligned} &\underset{x,\gamma}{\textbf{minimizar}}\ \frac{1}{2}\|x\|^2 \\ &s.a \\ &\quad y_i\left(x.s_i-\gamma\right)\geq 1, \quad i\in\{1,2,...,l\} \end{aligned} \tag{3.17}$$

onde l representa o número de elementos do conjunto de treinamento e y_i o rótulo do elemento, 1 para os membros da classe A+ e -1 para os membros da classe A- .

Pode ser observado que esta metodologia representa um problema de otimização quadrática. Usando a função *Lagrangeana,* esse problema de otimização pode ser transformado em uma formulação dual, onde a função objetivo depende unicamente do conjunto de multiplicadores de Lagrange. Esta formulação é dada por (PLATT, 1998):

$$\underset{\alpha}{\textbf{maximizar}}\ X(\alpha) = -\sum_{i=1}^{l}\alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (s_i . s_j)$$

$$s.a. \qquad (3.18)$$

$$\sum_{i=1}^{l} y_i \alpha_i = 0$$

$$\alpha_i \geq 0, \quad \forall i = 1, ..., l$$

Nesta formulação α é um vetor com l componentes, onde cada uma representa um elemento do conjunto de treinamento, ou seja, a componente α_i corresponde ao elemento (s_i, y_i).

Existe uma relação um para um entre cada multiplicador de Lagrange e cada elemento do conjunto de treinamento. Uma vez que os multiplicadores de Lagrange são calculados, o vetor normal x e γ podem ser determinados a partir da seguinte relação:

$$x = \sum_{i=1}^{l} y_i \alpha_i s_i, \qquad (3.19)$$

$$\gamma = x.s_k - y_k \quad para\ algum\ \alpha_k > 0.$$

Como x pode ser calculado, pela equação (3.18), a partir do conjunto de treinamento, o custo computacional para calcular um SVM linear é constante e está diretamente relacionado ao número de vetores suporte.

3.2.5.2 – Problemas não linearmente separáveis

Como na maioria dos casos práticos não é possível se verificar uma separabilidade linear dos dados, a extensão do modelo se mostra bastante importante na aplicabilidade da metodologia SVM.

Em 1995, Cortes e Vapnik (CORTES *et al.*, 1995) sugeriram uma modificação na formulação do problema de otimização anterior (3.17) permitindo que um ponto seja classificado de forma errada, mas penalizando essas ocorrências. Dentro desse referencial, o novo problema é dado por:

$$\begin{aligned} &\underset{w,b}{\textbf{minimizar}}\ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i \\ &s.a \\ &\quad y_i(w.x_i - b) \geq 1 - \xi_i, \quad i \in \{1,2,...,l\} \\ &\quad \xi_i \geq 0 \end{aligned} \tag{3.20}$$

onde ξ_i é uma variável de folga que permite a observação i violar a margem e C é o fator de penalidade, que estabelece um compromisso entre o erro de treinamento e a margem. Esse parâmetro C pode ser visto também como um parâmetro que penaliza os pontos que violam a margem.

Quando este novo problema é transformado em sua forma dual, somente a restrição de desigualdade do problema, em relação ao problema (3.18), é modificada e a variável ξ_i, que possibilita que uma observação viole a margem, não aparece nessa formulação. O novo problema é dado por (PLATT, 1998):

$$\begin{aligned} &\underset{\alpha}{\textbf{maximizar}}\ X(\alpha) = -\sum_{i=1}^{l}\alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (s_i, s_j) \\ &s.a. \\ &\quad \sum_{i=1}^{l} y_i \alpha_i = 0 \\ &\quad 0 \leq \alpha_i \leq C, \quad \forall i = 1,...,l. \end{aligned} \tag{3.21}$$

A formulação SVM pode ser generalizada também para utilizar separadores não lineares (BURGES, 1998). A classificação de uma nova observação desconhecida s, através de um classificador não linear, é feita explicitamente pelos multiplicadores de Lagrange, como segue:

$$u = \sum_{i=1}^{l} y_i \alpha_i K(s_i, s) - \gamma \tag{3.22}$$

onde K, denominada *função kernel*, indica a similaridade ou distância entre o vetor de entrada s e o vetor do conjunto de treinamento s_i. Dessa forma, caso $u \geq 1$ sabe-se que x pertence à classe 1, por outro lado, caso $u \leq -1$ a observação s será classificada como sendo da classe -1.

Alguns exemplos de funções *kernel* incluem Gaussianas, Polinomiais, funções de ativação do tipo sigmoidal. Se K é linear, então se tem o SVM linear original (3.16).

Os multiplicadores de Lagrange α_i continuam sendo calculados por um problema de otimização quadrática. Apesar da não-linearidade alterar ligeiramente a forma quadrática, a função objetivo dual continua sendo quadrática em α:

$$\begin{aligned} &\underset{\alpha}{\textbf{maximizar}}\ X(\alpha) = -\sum_{i=1}^{l}\alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(s_i, s_j) \\ &s.a. \\ &\qquad \sum_{i=1}^{l} y_i \alpha_i = 0 \\ &\qquad 0 \le \alpha_i \le C, \quad \forall i = 1, ..., l. \end{aligned} \tag{3.23}$$

Para que o problema anterior seja positivo definido, a função *kernel* deverá obedecer às condições de Mercer (BURGES, 1998).

As condições de *Karush-Kuhn-Tucker* (KKT) são condições necessárias e suficientes para um ponto de um problema quadrático positivo definido, pois o problema é convexo. As condições de KKT para o problema (3.21) são particularmente simples. O problema é solucionado quando, para todo *i*:

$$\begin{aligned} &\alpha_i = 0 \Leftrightarrow y_i u_i \ge 1, \\ &0 < \alpha_i < C \Leftrightarrow y_i u_i = 1, \\ &\alpha_i = C \Leftrightarrow y_i u_i \le 1. \end{aligned} \tag{3.24}$$

onde $u_i = \pm 1$ é a saída do SVM para o *i*-ésimo exemplo do conjunto de treinamento.

4.1 – Introdução

No capítulo anterior foram apresentadas diversas técnicas de classificação e agrupamento para categorização dos dados. Dentre os problemas apresentados, pode-se observar que a grande maioria de suas formulações é de natureza não diferenciável e não convexa.

No presente trabalho, será apresentada uma alternativa ao problema SVM apresentado no capítulo 3. Neste capítulo, será desenvolvida uma metodologia que transforma a formulação do problema SVM original em uma formulação mais simples, em um espaço de menor dimensão, que seja diferenciável C^{∞} e convexa. Na nova formulação uma seqüência de subproblemas diferenciáveis são solucionados aproximando-se do problema original.

Cada subproblema possui a forma padrão dos problemas de otimização, ou seja, possui uma função objetivo e um conjunto de m restrições de igualdade e p restrições de desigualdade, problema (4.1). Esses problemas são transformados intrinsecamente em problemas irrestritos através da técnica de penalização hiperbólica (XAVIER, 1982). Assim, tem-se um problema de otimização irrestrito e completamente diferenciável. A vantagem dessa metodologia é que cada um dos subproblemas pode ser solucionado, de forma mais simples, por métodos poderosos que utilizam informações da derivada, como gradiente conjugado, quase-Newton, dentre outros.

$$\begin{aligned} &\textbf{minimizar } f(x) \\ &s.a \quad g_i(x) \geq 0, \quad i = 1, ..., m \\ &\qquad h_j(x) = 0, \quad j = 1, ..., p \end{aligned} \tag{4.1}$$

4.2 – Definição do Problema

Existem várias formulações para os problemas de classificação. No capítulo anterior foi definido o problema de máquina de vetores suporte descrito por Vapnik, que foi o pioneiro na proposição desse problema. Para o desenvolvimento da nova metodologia que será apresentada neste capítulo será adotada como base uma outra

formulação do problema SVM definida por Mangasarian (MANGASARIAN *et al.*, 1999).

4.2.1 – Formulação SVM de Mangasarian

Como descrito no capítulo 3, a margem definida pelo hiperplano separador é igual a $2/\|w\|$. O objetivo do problema SVM é maximizar essa margem gerando um classificador com maior poder de generalização. Como maximizar a margem é o mesmo que minimizar o seu inverso, o problema SVM minimiza a função objetivo $\|w\|^2/2$.

A formulação do problema SVM de Mangasarian é equivalente a desenvolvida por Vapnik descrita na equação (3.17). Para os problemas de classificação, nos quais o conjunto de treinamento possui m pontos definidos no $\Re^n$ e esses são linearmente separáveis, Mangasarian propôs a seguinte formulação:

$$\begin{aligned} &\underset{(w,\gamma)\in\Re^{n+1+m}}{\textbf{minimizar}} \quad \tfrac{1}{2}w'w \\ &\qquad s.a \quad D(Aw-e\gamma)\geq e. \end{aligned} \tag{4.2}$$

Sendo o conjunto de observações de treinamento $s_i, i=1,...,m,$ representado pela matriz $A(m\times n)$, D é uma matriz diagonal $(m\times n)$ com diagonal +1 ou -1. Caso a observação s_j pertença a classe +1 então $D_{jj}=+1$, caso contrário $D_{jj}=-1$. A variável w é a normal ao hiperplano separador e γ é o deslocamento deste hiperplano em relação à origem.

A superfície de separação linear é dada pelo hiperplano

$$s'w=\gamma \tag{4.3}$$

e os hiperplanos que limitam as classes, ou seja, os hiperplanos que estão nos limites das classes +1 e −1 são definidos, respectivamente, por:

$$\begin{aligned} s'w-\gamma&=+1 \\ s'w-\gamma&=-1, \end{aligned} \tag{4.4}$$

onde a distância entre esses hiperplanos determina a margem entre as duas classes.

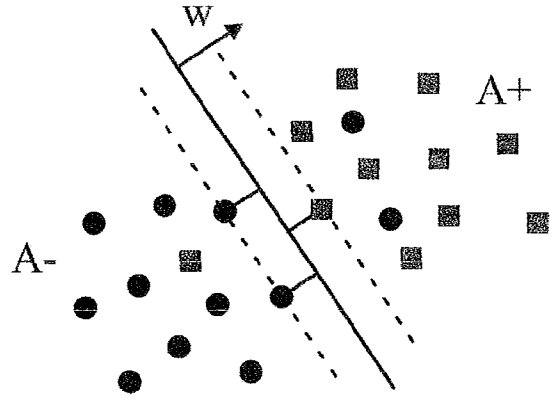


Figura 4.1 - Problema não linearmente separável

Como a maior parte dos problemas de classificação são não linearmente separáveis, adiciona-se uma variável de folga ξ no problema (4.2), de forma que seja permitido que uma observação viole a margem produzida pelo classificador. Assim, com o uso da variável de folga, as observações responsáveis por tornar o problema não separável ficam classificadas de forma errônea, como pode ser observado na Figura 4.1. O novo problema fica então definido por:

$$\begin{aligned} \underset{(w,\gamma)\in\Re^{n+1+m}}{\textbf{minimizar}} \quad & C\sum_{i=1}^{m}\xi_i + \tfrac{1}{2}w'w \\ s.a \quad & y_i\left(w.s_i - \gamma\right) + \xi_i \geq e \\ & \xi_i \geq 0, \quad i \in \{1,2,...,m\}. \end{aligned} \tag{4.5}$$

Para o problema anterior, no qual os dados não são linearmente separáveis, os hiperplanos da equação (4.6) determinam os limites das classes definindo uma *margem suave* (*soft margin*). Essa margem está diretamente relacionada à não negatividade da variável de folga ξ.

$$\begin{aligned} s_i'w - \gamma + \xi_i \geq +1, \quad & para \;\; y_i = +1, \\ s_i'w - \gamma - \xi_i \leq -1, \quad & para \;\; y_i = -1. \end{aligned} \tag{4.6}$$

4.2.2 – Formulação SVM adotada

Neste trabalho, será adotada a formulação a seguir especificada. Seja $S = \{s_1, s_2, ..., s_m\}$ um conjunto de m observações, onde cada observação genérica $s_j \in \Re^n$, $j = 1, ..., m$. Seja um conjunto $J = \{1, 2, ..., m\}$ tal que $J = J_1 \cup J_2$, onde J_1 é o conjunto das observações pertencentes à classe 1 e J_2 é o conjunto de observações pertencentes à classe -1. Ou seja, se $s_j \in J_1$ a observação pertence a classe 1 e caso $s_j \in J_2$ será uma observação pertencente à classe 2. Assim, $J_1 \cap J_2 = \emptyset$.

O objetivo deste problema é encontrar um hiperplano $x \in \Re^n$ que separe os dados das duas classes da melhor forma possível segundo um critério particular. Especificamente, pretende-se obter um hiperplano tal que a margem de separação entre os dados seja a maior possível.

Seja

$$Z_i = \underset{j \in J_i}{\text{mínimo}} \; x s_j + \gamma, \qquad i = 1, 2 \tag{4.7}$$

a distância mínima entre a classe i e o hiperplano definido por x. Como deseja-se maximizar a margem, um novo problema de otimização pode ser definido da seguinte forma

$$\begin{aligned} &\text{minimizar} - (Z_1 + Z_2) \\ &s.a \\ &\quad Z_1 = \underset{j \in J_1}{\text{mínimo}} \left(x s_j + \gamma \right) \\ &\quad Z_2 = \underset{j \in J_2}{\text{mínimo}} \left(-\left(x s_j + \gamma \right) \right) \\ &\quad \|x\|_2 = 1 \end{aligned} \tag{4.8}$$

As restrições do problema (4.8), após uma simples manipulação, podem ser reescritas como

$$
\begin{aligned}
Z_1 &= \left[\underset{j \in J_1}{\text{mínimo}} \left(x\, s_j \right) \right] + \gamma \\
Z_2 &= \left[\underset{j \in J_2}{\text{mínimo}} \left(-x\, s_j \right) \right] - \gamma.
\end{aligned}
\tag{4.9}
$$

Pode ser observado que a variável γ é anulada pelas soma das restrições. Assim, desde que a variável γ entra na função objetivo com sinais contrários, o resultado é indiferente ao seu valor assumido. Como o papel de γ é completamente inócuo, o mesmo será retirado do problema (4.8) sem qualquer prejuízo. A partir da retirada da variável γ, chega-se a seguinte formulação equivalente, com dimensão $(n+2)$:

$$
\begin{aligned}
&\underset{x, Z_1, Z_2}{\text{minimizar}} - \left(Z_1 + Z_2 \right) \\
&s.a \\
&\quad Z_1 = \underset{j \in J_1}{\text{mínimo}} \left(x\, s_j \right) \\
&\quad Z_2 = \underset{j \in J_2}{\text{mínimo}} \left(-x\, s_j \right) \\
&\quad \left\| x \right\|_2 = 1
\end{aligned}
\tag{4.10}
$$

Para os problemas cujos dados do conjunto de treinamento são linearmente separáveis pode ser definida a seguinte relação entre a formulação proposta (4.8) e a formulação de Mangasarian (4.5)

$$
Z_1 + Z_2 = \frac{2}{\left\| w \right\|_2} \tag{4.11}
$$

$$
\gamma = \frac{Z_1 - Z_2}{2} \tag{4.12}
$$

$$
x = \frac{Z_1 + Z_2}{2} w \tag{4.13}
$$

onde w é o vetor normal ao hiperplano separador na formulação do Mangasarian.

As racionalidades dessas relações acima podem ser obtidas facilmente. Como $Z_1 + Z_2$ define a margem entre as classes, a relação (4.11) é dada diretamente. Tem-se que γ é o deslocamento do hiperplano separador à origem, portanto, com o auxílio da Figura 4.2, a relação (4.12) pode ser inferida. Pode ser observado que o vetor x, da

formulação proposta, e o vetor w, da formulação de Mangasarian, são vetores normais ao hiperplano. Entretanto, w não está normalizado, enquanto que $\|x\|_2 = 1$. Assim, pode-se afirmar que

$$x = \frac{w}{\|w\|_2} \tag{4.14}$$

e substituindo (4.11) em (4.14) tem-se a relação (4.13).

Assim, a partir das relações definidas acima, para o caso onde as classes são linearmente separáveis, obtém-se que o problema (4.8) é equivalente ao problema (4.5) e, conseqüentemente, suas soluções são iguais.

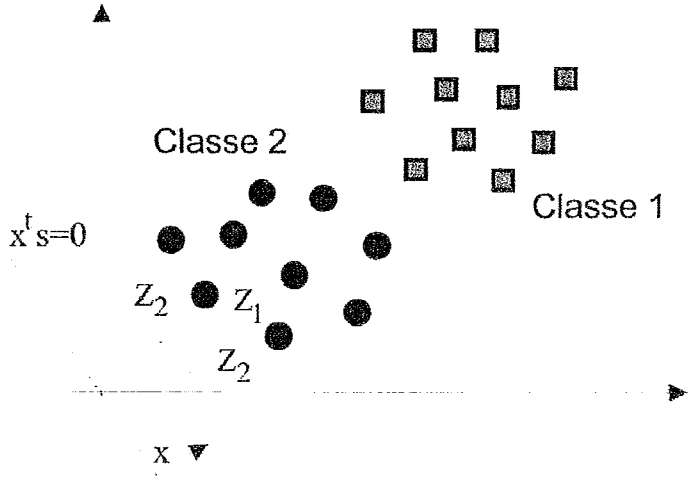


Figura 4.2 - Distância mínima entre a classe e o hiperplano

Para os problemas nos quais o conjunto de treinamento é formado por dados não linearmente separáveis a igualdade das soluções está diretamente relacionada ao valor do parâmetro C, definido nas formulações de Mangasarian e Vapnik.

Para essas duas formulações, quando o parâmetro C decresce, o termo de maximização da margem tem sua importância gradualmente crescente. Assim, conforme C decresce, o tamanho da margem aumenta (GUNN, 1998). Quando o parâmetro $C = 0$, toda a ênfase é dada ao termo de maximização da margem, e a solução produzida possui a máxima margem. Dessa forma, com $C = 0$, as soluções de Mangasarian e Vapnik e da formulação proposta continuam sendo iguais.

A Figura 4.3 tem o objetivo de ilustrar essa situação. Os três pontos extremos estão envolvidos com linhas tracejadas. Quando $C=0$, somente esses três pontos influenciam as soluções de Mangasarian e Vapnik e da formulação proposta.

Por outro lado, no caso em que $C \neq 0$ as soluções obtidas são diferentes. Essa diferença é produzida pela contribuição diferenciada dos pontos não-separáveis nos dois critérios. No critério de Mangasarian e Vapnik todos os pontos não separáveis contribuem na obtenção do hiperplano separador. Já na formulação proposta, somente os pontos mais não linearmente separáveis contribuem, ou seja, somente os pontos que mais violam a margem participam da definição do novo hiperplano.

A Figura 4.3 igualmente pode ser usada para ilustrar esse caso. A solução produzida pelas formulações de Mangasarian e Vapnik com $C \neq 0$ seriam influenciadas por todos os pontos com violação. Esses pontos são mostrados na figura por 4 círculos cheios e 5 quadrados cheios.

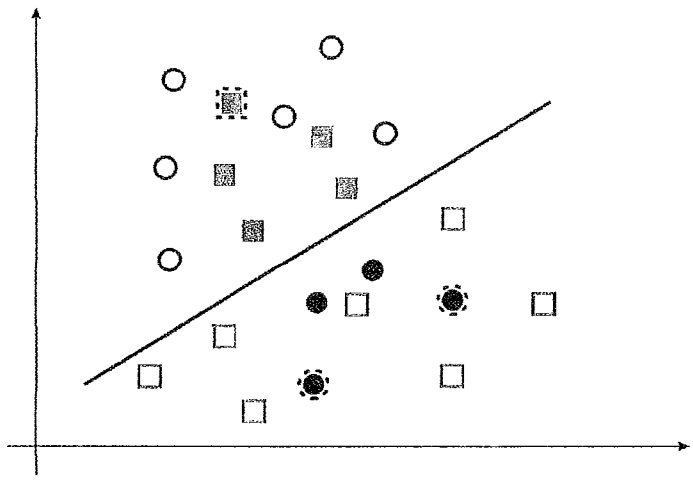


Figura 4.3 - Solução do problema não linearmente separável ($C=10^{-8}$)

4.3 – Transformação do Problema

O problema anterior apresenta uma dificuldade intrínseca em suas restrições, já que estas apresentam uma particularidade de serem não diferenciáveis. Buscando eliminar esta dificuldade será utilizada a técnica de suavização hiperbólica (XAVIER, 1982). Essa técnica tem por objetivo transformar equações não diferenciáveis em diferenciáveis. Assim, aplicando essa técnica no problema anterior, esse poderá ser resolvido por qualquer método de otimização que faça uso de derivadas primeiras ou

segundas. Esses métodos, como são amplamente difundido na literatura, têm desempenhos computacionais superiores, tanto analisando o critério de robustez (capacidade de produzir soluções corretas) como de eficiência (capacidade de produzir soluções rapidamente).

Entretanto, a transformação do problema (4.10) em um problema diferenciável requer algumas mudanças em sua definição. Neste tópico, o problema será preparado para posteriormente ser aplicada a técnica de suavização.

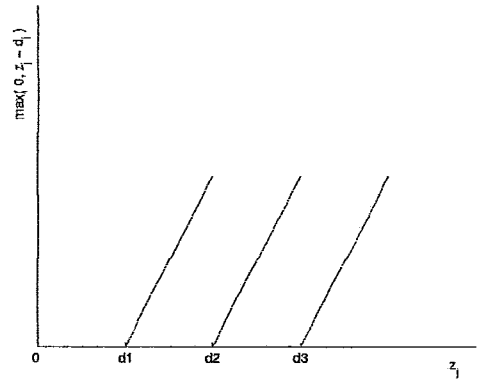


Figura 4.4 - Somatório da lado direito das restrições (4.17).

Primeiramente, considere a função φ definida da seguinte forma:

$$\varphi(y) = \max(0, y). \tag{4.15}$$

Observe que pelas restrições 1 e 2, Z_i, $i = 1,2$ representa a menor distância entre a respectiva classe e o hiperplano separador, como mostra a Figura 4.2. Assim, estas restrições podem ser reescritas na forma de desigualdades, transformando o problema original como segue:

$$\begin{aligned} &\underset{x, Z_1, Z_2}{\text{minimizar}} -(Z_1 + Z_2) \\ &s.a \\ &\quad Z_1 - s_j^t x \le 0, \quad j \in J_1 \\ &\quad Z_2 + s_j^t x \le 0, \quad j \in J_2 \\ &\quad \|x\|_2 = 1 \end{aligned} \tag{4.16}$$

Com o auxílio da função φ as restrições do problema (4.16) podem ser escritas na forma:

$$\begin{aligned} &\sum_{j\in J_1} \varphi\left(Z_1 - s_j^t x\right) = 0 \\ &\sum_{j\in J_2} \varphi\left(Z_2 + s_j^t x\right) = 0 \end{aligned} \tag{4.17}$$

Considerando a primeira restrição do problema (4.16) e assumindo $J_1 = \{1, 2, 3, ..., m_1\}$ e $d_1 < d_2 < \ldots < d_{m_1}$ com $d_j = s_j^t x,\ j \in J_1$, a Figura 4.4 ilustra os três primeiros somatórios do lado esquerdo dessa restrição como uma função de Z_1, como na figura anterior.

Assim, substituindo as restrições do problema (4.16) definidas em (4.17), respectivamente, obtém-se o problema transformado abaixo:

$$\begin{aligned} &\underset{x, Z_1, Z_2}{\text{minimizar}} -\left(Z_1 + Z_2\right) \\ &s.a \\ &\sum_{j\in J_1} \varphi\left(Z_1 - s_j^t x\right) = 0 \\ &\sum_{j\in J_2} \varphi\left(Z_2 + s_j^t x\right) = 0 \\ &\|x\|_2 = 1 \end{aligned} \tag{4.18}$$

4.4 – Suavização do Problema

Embora o problema (4.18) seja de dimensão muito menor, quando comparado ao problema (4.2) de Mangasarian, todas as suas restrições continuam sendo não diferenciáveis, o que torna a obtenção de sua solução, como dito anteriormente, altamente difícil. Neste tópico, o problema será suavizado de forma a obter um problema completamente diferenciável (XAVIER, 1982).

Sobre esse ponto de vista, considere a seguinte função:

$$\phi(y, \tau) = \left(y + \sqrt{y^2 + \tau^2}\right)\Big/2 \tag{4.19}$$

para $y \in \Re$ e $\tau > 0$.

Esta função possui as seguintes propriedades:

1. $\phi(y,\tau) > \varphi(y), \quad \forall \tau > 0$;
2. $\lim_{\tau \to 0} \phi(y,\tau) = \varphi(y)$;
3. $\phi(.,\tau)$ é uma função C^{∞} convexa e crescente.

Pela segunda propriedade da equação (4.19), pode ser observado que para $\tau = 0$ a função $\phi(y,\tau)$ é equivalente à função $\varphi(y)$, visto que para $y \leq 0$ tem-se $\phi(y,0) = 0$ e para $y > 0$ obtém-se $\phi(y,0) = y$. Entretanto, para $\tau > 0$, como descrito na terceira propriedade da função, a função $\phi(.,\tau)$ é C^{∞}, enquanto que a função $\varphi(y)$ diferenciadamente é não diferenciável.

A Figura 4.5 exibe simultaneamente o gráfico dos três primeiros somatórios da primeira restrição do problema (4.18) e suas aproximações suavizadas dadas pela função $\phi(y,\tau)$ da equação (4.19).

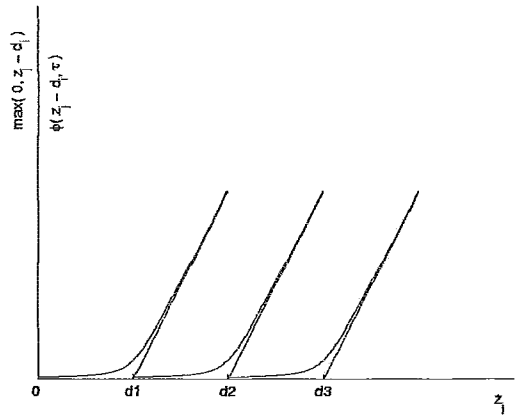


Figura 4.5 - Somatório original e suavizado das restrições do problema.

Substituindo a função $\varphi(y)$, não diferenciável, pela função $\phi(y,\tau)$ e incluindo uma perturbação $\phi(.,\tau)$ no lado direito das restrições é obtido um problema completamente diferenciável, abaixo definido:

$$
\begin{aligned}
&\underset{x, Z_1, Z_2}{\text{minimizar}} -\left(Z_1 + Z_2\right) \\
&s.a \\
&\sum_{j \in J_1} \phi\left(Z_1 - s_j^t x,\ \tau\right) \leq \varepsilon \\
&\sum_{j \in J_2} \phi\left(Z_2 + s_j^t x,\ \tau\right) \leq \varepsilon \\
&\|x\|_2 = 1
\end{aligned}
\tag{4.20}
$$

4.5 – Reformulação do problema

Definindo os parâmetros τ e ε no problema (4.20) obtém-se o problema (4.18). Assim, a solução do problema (4.18) pode ser obtida resolvendo várias formulações do problema (4.20) decrescendo os parâmetros $\tau \to 0$ e $\varepsilon \to 0$. Definindo-se um processo iterativo no qual em cada iteração os parâmetros τ e ε são reduzidos, tendendo a zero, tem-se a garantia de obter o problema original.

Sendo a função $\phi(.,\tau)$ crescente e convexa (propriedade 3), as restrições do problema (4.20) serão certamente ativas para qualquer vetor x. Assim, o problema anterior será equivalente ao problema de dimensão $(n+2)$ descrito abaixo.

$$
\begin{aligned}
&\underset{x, Z_1, Z_2}{\text{minimizar}} -\left(Z_1 + Z_2\right) \\
&s.a \\
&h_1\left(Z_1, x\right) = \sum_{j \in J_1} \phi\left(Z_1 - s_j^t x,\ \tau\right) - \varepsilon = 0 \\
&h_2\left(Z_2, x\right) = \sum_{j \in J_2} \phi\left(Z_2 + s_j^t x,\ \tau\right) - \varepsilon = 0 \\
&\|x\|_2 = 1
\end{aligned}
\tag{4.21}
$$

Entretanto, o problema (4.21) possui uma estrutura separável, pois cada variável Z_1 e Z_2 aparece em somente uma restrição de igualdade. Desta forma, já que as derivadas parciais de cada restrição $h_1\left(Z_1, x\right)$ e $h_2\left(Z_2, x\right)$ em relação a Z_1 e Z_2, respectivamente, não é igual a zero, é possível usar o Teorema da Função Implícita para calcular Z_1 e Z_2 como função de x.

Assim, o problema

$$\begin{aligned} &\min f(x) = -\left(Z_1(x) + Z_2(x)\right) \\ &s.a \\ &\quad \|x\|_2 = 1 \end{aligned} \tag{4.22}$$

é obtido, onde $Z_1(x)$ e $Z_2(x)$ resultam do cálculo das raízes das equações a seguir.

$$h_i(Z_i, x) = \sum_{j \in J_1} \phi\left(Z_i - s_j^t x,\ \tau\right) - \varepsilon = 0 \qquad i = 1, 2. \tag{4.23}$$

Observando a terceira propriedade da função de suavização hiperbólica, cada termo ϕ é intrinsecamente crescente com a variável Z_i. Assim, a equação tem somente uma raiz.

Novamente, pelo Teorema da Função Implícita, as funções $Z_i(x)$, $i = 1, 2$, têm todas as derivadas em relação ao vetor x. Logo, é possível calcular o gradiente da função objetivo do problema (4.22) como segue:

$$\nabla f(x) = \sum_{i=1}^{2} \nabla Z_i(x) \tag{4.24}$$

onde

$$\nabla Z_i(x) = -\nabla h_i(Z_i, x) \Big/ \frac{\partial h_i(Z_i, x)}{\partial Z_i}. \tag{4.25}$$

Primeiramente, deve ser observado que a função objetivo do problema (4.22) é uma função homogênea de grau um do vetor x, com norma Euclidiana $\|x\|_2$. Então, é possível substituir a restrição de igualdade do problema (4.22) por duas restrições de desigualdade:

$$
\begin{aligned}
g_1(x) &= u - \|x\|_2 \geq 0, \\
g_2(x) &= \|x\|_2 - l \geq 0,
\end{aligned}
\tag{4.26}
$$

para todo $0 < l < u$. Com essa substituição acima, obtém-se o problema:

$$
\begin{aligned}
&\underset{x}{\text{minimizar}}\ f(x) = -\left(Z_1(x) + Z_2(x)\right) \\
&s.a \\
&\quad g_1(x) = u - \|x\|_2 \geq 0, \\
&\quad g_2(x) = \|x\|_2 - l \geq 0.
\end{aligned}
\tag{4.27}
$$

Se o problema for linearmente separável, a primeira desigualdade será ativa, e, já que a segunda é não ativa, desprezando-a, tem-se assim um problema essencialmente convexo. Para os problemas não linearmente separáveis a segunda desigualdade é que estará ativa, obtendo-se um problema de natureza não convexa.

Finalmente, o problema (4.27) definido pela suavização hiperbólica é um problema de programação não-linear com restrições de desigualdade, portanto, poderia ser solucionado por qualquer método existente na literatura de otimização. Será definida no próximo tópico a metodologia de penalização hiperbólica, que será a utilizada na resolução do problema definido anteriormente.

A vantagem dessa escolha é que há uma natural articulação entre os parâmetros de suavização e de penalização, como será apresentado.

4.6 – Penalização Hiperbólica

O Método de Penalização Hiperbólica, definido de maneira mais completa em (XAVIER, 1982), é utilizado para solucionar problemas gerais de otimização sujeito a restrições de desigualdade. Esse método é uma alternativa aos demais métodos difundidos na literatura para a solução de problemas de programação não linear. Em geral esses problemas estão na forma:

$$
\begin{aligned}
&\textit{minimizar} \quad f(x) \\
&\textit{s.t.} \quad g_i(x) \geq 0,\ i = 1, \ldots, m
\end{aligned}
\tag{4.28}
$$

onde $f:\Re^n \to \Re$ e $g_i:\Re^n \to \Re,\ i=1,...,m$.

Essa metodologia incorpora as restrições do problema de otimização à função objetivo utilizando uma função penalidade. Assim, é gerado um problema irrestrito que tem uma função objetivo modificada ou penalizada, como segue:

$$\text{minimizar} \quad f(x)+\sum_{i=1}^{m}\overline{P}\left(g_i(x)\right) \tag{4.29}$$

onde o segundo termo é o termo de penalização.

A função de penalização é pertencente a classe de funções C^{∞} e é definida por:

$$\overline{P}(y,\alpha,\rho)=-(\frac{1}{2}\tan\alpha)y+\sqrt{(\frac{1}{2}\tan\alpha)^2y^2+\rho^2}\,, \tag{4.30}$$

onde $\alpha\in[0,\pi/2)$ e $\rho>0$. Essa função, como mostra a Figura 4.6, possui uma assíntota horizontal e uma inclinada com ângulo α, interceptando em ρ com o eixo das ordenadas.

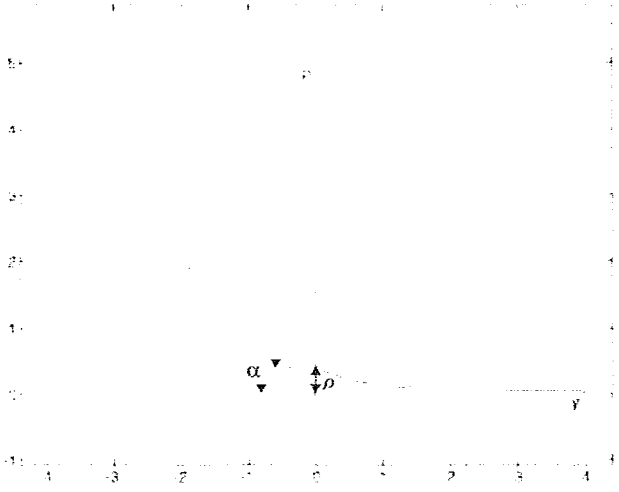


Figura 4.6 – Função de Penalização Hiperbólica

De forma alternativa, a função de penalização hiperbólica pode ser definida por:

$$\overline{P}(y,\lambda,\rho)=-\lambda y+\sqrt{\lambda^2y^2+\rho^2} \tag{4.31}$$

com $\lambda \geq 0$ e $\rho \geq 0$.

O novo problema que será otimizado possui a nova função objetivo modificada, sendo formulado como segue:

$$F(x, \lambda^k, \rho^k) = f(x) + \sum_{i=1}^{m} P(g_i(x), \lambda^k, \rho^k). \tag{4.32}$$

É importante observar que fixando o parâmetro $\rho = 0$, a resolução do problema (4.32) é equivalente à resolução do problema original (4.28), conforme demonstrado em (XAVIER, 1982). Assim, para solucionar o problema (4.32) é gerada uma seqüência de subproblemas intermediários, $k = 1, 2, 3, \ldots$, de forma que os parâmetros λ e ρ decresçam a cada iteração k.

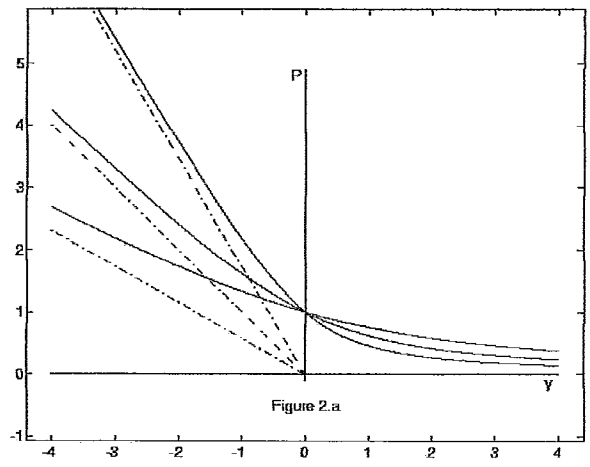


Figura 4.7 - Variação do parâmetro λ.

Cada um dos parâmetros da função penalidade trabalha em uma região do espaço onde o problema está definido. O parâmetro λ trabalha sobre a região inviável, de forma que o seu acréscimo eleva a penalidade sobre os pontos inviáveis, levando-os para a região viável, Figura 4.7. Por outro lado, o decréscimo do parâmetro τ faz com que o valor da penalidade decresça assintoticamente a zero, conforme na Figura 4.8.

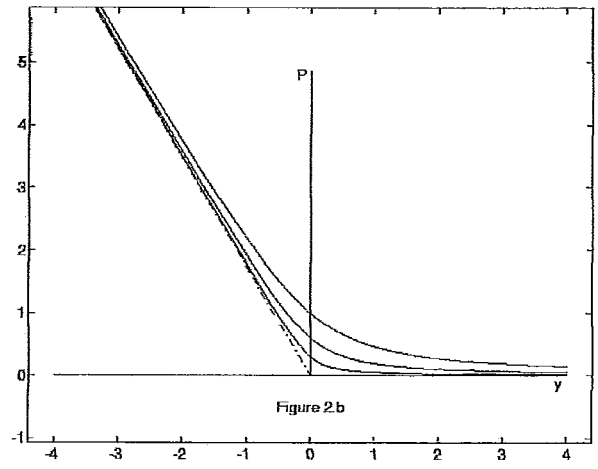


Figura 4.8 - Variação do parâmetro ρ.

A maior parte dos métodos de penalização utilizam apenas um parâmetro. Já a penalização hiperbólica, como descrito anteriormente, trabalha com dois parâmetros de penalização. Assim, o algoritmo da penalização hiperbólica, descrito a seguir, deve, naturalmente, manipular esses dois parâmetros.

Algoritmo de Penalização Hiperbólica

1: Faça $k = 0$ e x^0, $\lambda^1 > 0$, $\rho^1 > 0$.

2: k := k + 1 e resolva o problema: **minimizar**$_x$ $F(x, \lambda^k, \rho^k)$

3: Se x^k é um ponto inviável então faça

$$\lambda_i^{k+1} := r\lambda_i^k, \quad para\ r > 1$$

e retorne ao passo 2.

4: Senão faça

$$\rho_i^{k+1} = q\rho_i^k, \quad para\ 0 < q < 1$$

e retorne ao passo 2.

De acordo com a publicação original do método de penalização hiperbólica (XAVIER, 1982) o algoritmo funciona da seguinte maneira:

"A seqüência de subproblemas é obtida pela variação controlada dos dois parâmetros, λ e ρ, em duas diferentes fases do algoritmo. Inicialmente, se aumenta o parâmetro λ, causando um aumento significativo na penalização fora da região viável e, ao mesmo tempo, uma redução significativa na penalização para os pontos

dentro da região viável. Esse processo continua até que se obtenha um ponto viável. Daí em diante, mantém-se λ *constante e se diminui* ρ *seqüencialmente. Dessa maneira, a penalização interna fica cada vez mais irrelevante, mantendo o mesmo nível de proibição na região externa.*"

Articulando a suavização hiperbólica com a técnica de penalização hiperbólica descrita anteriormente, o problema de otimização (4.27) pode ser redefinido completamente irrestrito como segue:

$$\min F(x,\lambda,\rho) = -\left(Z_1(x)+Z_2(x)\right)+\sum_{i=1}^{2} P\left(g_i(x),\lambda,\rho\right) \tag{4.33}$$

onde $P\left(y,\lambda,\rho\right) = -y\lambda+\left((y\lambda)^2+\rho^2\right)^{1/2}$.

Desta maneira, o problema (4.33) é facilmente resolvido utilizando-se qualquer método que se baseie em informações das derivadas de primeira ordem da função ou segunda ordem, que são mais robustas, como registrado amplamente na literatura, por exemplo (MINOUX, 1986). É importante observar que o problema (4.27) é definido somente no espaço de dimensão n, dessa forma, o conjunto de transformações efetuadas gerou um problema de otimização com a mais completa independência do número de observações m.

4.7 – Algoritmo SHSVM simplificado

Primeiro passo:

- Escolha os valores iniciais para x^0, τ^1, ρ;
- Escolha um valor fixo para o parâmetro λ da penalização hiperbólica suficientemente alto;
- Escolha os valores: $0<q_1<1$, $0<q_2<1$, $0<q_3<1$. Faça $k=0$.

Passo principal: Repita até obter o Critério de Parada

- Resolva o problema (4.33) com $\tau=\tau^k$ e $\varepsilon=\varepsilon^k$, começando no ponto inicial x^{k-1}, sendo x^k a solução obtida.
- Faça $\tau^{k+1}=q_1\tau^k$, $\varepsilon^{k+1}=q_2\varepsilon^k$, $\rho^{k+1}=q_3\rho^k$. Faça $k=k+1$.

A solução do problema SVM é obtida resolvendo uma seqüência infinita de problemas de otimização (passo principal), no presente caso, uma seqüência de subproblemas de minimização irrestritos.

Note que o algoritmo faz com que τ e ε se aproxime de 0, tornando o problema (4.33) equivalente ao problema original.

Capítulo 5 – Implementação e Experimentos Computacionais

Esta dissertação contempla, como descrito nos capítulos 3 e 4, dentro do escopo de Mineração de Dados, a questão de classificação na abordagem de Máquina de Vetores Suporte (SVM) proposta por Vapnik (VAPNIK, 1995).

Com o objetivo de resolver um modelo matemático associado à metodologia SVM foi apresentada uma nova proposta metodológica denominada SHSVM. A metodologia SHSVM tem toda a potencialidade de poder ser utilizada no equacionamento de problemas associados à Mineração de Dados de grandes bases de dados. Por exemplo, dados bancários, geográficos, de marketing ou de saúde, dentre outros.

Com o objetivo de fazer uma demonstração prática do uso do SHSVM serão apresentados a seguir os resultados computacionais obtidos na resolução de problemas canônicos de classificação sobejamente utilizados na literatura. Esses problemas teste foram obtidos no principal repositório disponível na *web* (ASUNCION *et al.*, 2007).

Os resultados obtidos foram comparados com o principal software de classificação utilizado atualmente, o *SVMLight* (JOACHIMS, 1998). Na execução desse software, foram utilizados os valores padrão das variáveis do mesmo, exceto no caso não linearmente separável, onde utilizou-se $C=10^{-5}$ para que a margem não fosse amplamente violada.

No algoritmo proposto, conforme descrito no capítulo 4, a solução é obtida através da resolução de uma seqüência de subproblemas irrestritos. As minimizações irrestritas foram realizadas por meio de um algoritmo Quase-Newton, com fórmula de atualização BFGS, disponibilizado pela Harwell Library (MURPHY *et al.*, 1992). Toda a implementação foi codificada na linguagem Fortran 77, e compilado utilizando-se o compilador DIGITAL Visual Fortran versão 6.0A. Os experimentos numéricos foram realizados em um PC, AMD Sempron 1.67 GHz, 1,00 Gb de memória RAM.

5.1 – Implementação

O primeiro passo do algoritmo é determinar o ponto inicial e a os parâmetros iniciais para o processo de otimização. Existem vários critérios para a escolha do ponto inicial, dentre eles, o mais utilizado é determinar o ponto inicial através dos centros de gravidade das duas classes. Assim, o hiperplano inicial é obtido calculando-se a metade da distância entre os centros de gravidade de cada umas das classes, como na Figura 5.1.

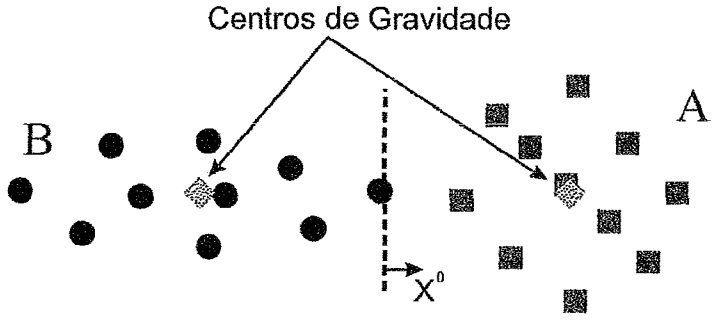


Figura 5.1 - Centros de gravidade e hiperplano inicial

Como observado no capítulo 4, que descreve a metodologia utilizada para a solução do problema de classificação, existe um conjunto de sete parâmetros associados ao algoritmo SHSVM. Os parâmetros τ e ε são do algoritmo de suavização hiperbólica, os parâmetros λ e ρ do algoritmo de penalização hiperbólica e os parâmetros q_1, q_2, q_3 são fatores de redução vinculados a cada um dos parâmetros anteriores. É natural se imaginar que esses sete parâmetros devam ser especificados de maneira harmônica para a obtenção de bons resultados.

A harmonização dos parâmetros é a principal dificuldade encontrada durante a implementação computacional. Trata-se de um trabalho árduo de tentativa e erro, na busca por intervalos numéricos que definam a melhor conformação dos parâmetros.

Em virtude da complexidade na definição dos parâmetros, será feita a seguir uma explanação sobre o papel e o funcionamento de cada um deles.

O parâmetro τ está diretamente relacionado ao nível de suavização do problema e sua variação é exibida, de forma ilustrativa, na Figura 5.2. Como pode ser observado, quanto menor for esse parâmetro, melhor será a aproximação da função suavizada à função original. Assim, para valores muito pequenos, o problema suavizado tem um comportamento nervoso próximo ao ponto de não diferenciabilidade, associado à variação muito forte da derivada primeira, ou seja, aos valores muito grandes da

derivada segunda. Por outro lado, a escolha de valores grandes implica em um distanciamento excessivo do problema original.

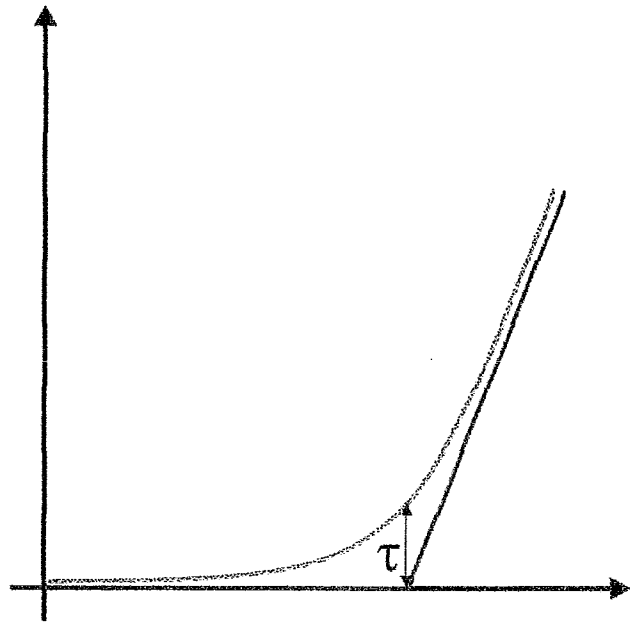


Figura 5.2 – Parâmetro τ.

O parâmetro de tolerância ε, relacionado à suavização hiperbólica, está naturalmente ligado ao parâmetro τ, vide equação (4.20). A manipulação desse parâmetro reflete na rigidez do hiperplano, de forma que o acréscimo de ε proporciona uma certa elasticidade ao hiperplano, pois um número maior de pontos interferirá de forma mais significativa na sua definição. A análise da Figura 5.3 possibilita uma idealização desse comportamento.

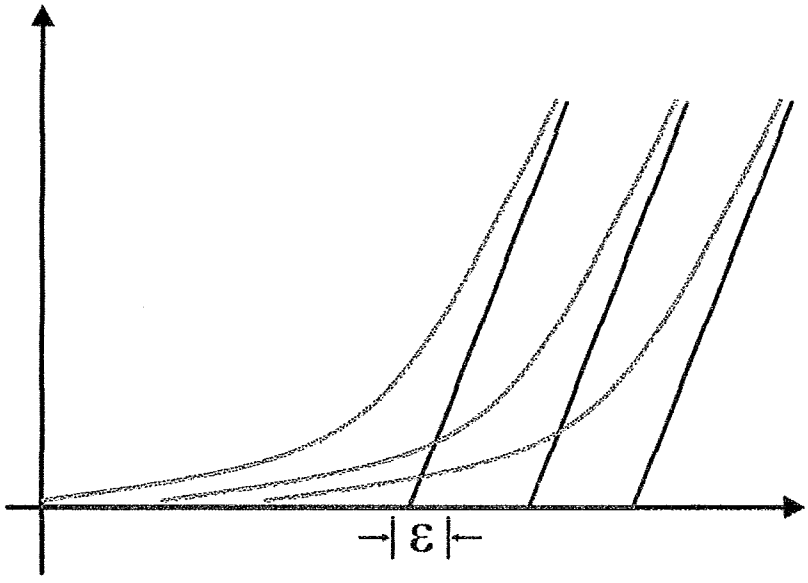


Figura 5.3 – Parâmetro ε.

O parâmetro λ da penalização hiperbólica está mais associado à penalização da violação das restrições, ou seja, produz mais efeito fora da região viável.

O parâmetro ρ da penalização hiperbólica está associado à relaxação das restrições e produz um maior efeito dentro da região viável. Assim, quando $\rho = 0$, as duas restrições, definidas em (4.20), são obedecidas rigorosamente.

Como dito anteriormente, a definição mais adequada dos parâmetros do algoritmo é específica para cada problema, onde a quantidade de observações, a dimensão do espaço de entrada e a distribuição das observações nesse espaço são fatores que influenciam fortemente essa definição. Cada problema apresenta uma melhor performance para um determinado conjunto específico de parâmetros, ou seja, não é possível determinar *a priori* uma conformação ideal que produza o melhor resultado.

Os experimentos mostraram que o parâmetro τ pode ser definido de acordo com a dimensão do problema, sendo adequado se tomar um valor inicial de τ no intervalo $\|x\|_2/\sqrt{n} \leq \tau \leq \|x\|_2/n$.

O parâmetro de tolerância ε da suavização tem uma forte associação com o outro parâmetro de suavização τ, conforme equação (4.20). Uma relação que se mostrou apropriada foi tomar $\tau/2 \leq \varepsilon \leq 10\tau$.

O valor do λ foi escolhido simplesmente para produzir ponto viável já na primeira iteração. Em termos práticos, implica em o algoritmo de penalização

hiperbólica só executar a segunda fase do mesmo. O valor suficientemente alto para tal propósito foi $\lambda = 1$.

Como dito anteriormente o parâmetro ρ da penalização hiperbólica também está diretamente ligado à τ. Os valores iniciais mais adequados para esse parâmetro da penalização hiperbólica foram obtidos fixando $\rho = \tau^2$.

A suavização hiperbólica por ser uma técnica que implica na resolução de uma seqüência de subproblemas, carece dos parâmetros q_1 e q_2, denominados por *fatores de redução*, responsáveis por aproximar, a cada iteração, o subproblema suavizado solucionado do problema original. Como visto no capítulo 4, a redução dos parâmetros τ e ε aproxima o problema suavizado ao problema original.

Analogamente, o método da penalização hiperbólica carece de um fator q_3 para redução do parâmetro ρ, para que seja provida a obediência rigorosa às restrições.

Nos experimentos computacionais, demonstrou-se eficaz a escolha $q_1 = q_2 = q_3$. A manipulação dos fatores de redução tem dois compromissos. Um valor excessivo para esses fatores prejudica a continuidade harmoniosa dos processos aproximativos. Essa falta de harmonia será refletida em iterações intermediárias mais demoradas. Por outro lado, valores pequenos aumentam desnecessariamente o número total de iterações. Os valores mais adequados para os problemas testados se encontram dentro da faixa $1.2 \leq q \leq 2.0$.

5.2 – Resultados computacionais

Essa seção realiza procedimentos de Mineração de Dados via classificação com enfoque SVM. Como dito no capítulo 2, o processo de Mineração de Dados depende, fundamentalmente, da etapa de extração de padrões. Nessa etapa, são utilizados algoritmos de agrupamento, seleção de características e classificação, entre outros. O destaque dado ao algoritmo de classificação via SVM se dá devido ao poder de generalização do método. Pois, a busca pela maximização da margem entre as classes leva, intuitivamente, a um classificador com maior probabilidade de acerto quando aplicado a dados novos.

Para comprovação da eficácia e robustez da metodologia apresentada neste trabalho, foram realizados experimentos com algumas das bases de dados mais

utilizadas na literatura. Essas bases foram obtidas a partir do repositório disponível na *web* em (ASUNCION *et al.*, 2007).

Para que seja possível uma análise visual da solução obtida pelo SHSVM, foi selecionada a base de dados Iris. Essa base de dados foi adotada por ter poucas características e por ser amplamente utilizada na literatura.

Esse problema possui três classes: Iris Setosa, Iris Versicolor e Iris Virginica. O seu conjunto de treinamento é constituído por um total de 150 observações, onde cada observação possui quatro atributos. Dois atributos são referentes ao comprimento e a largura das sépalas e os outros dois atributos são relacionados ao comprimento e a largura das pétalas.

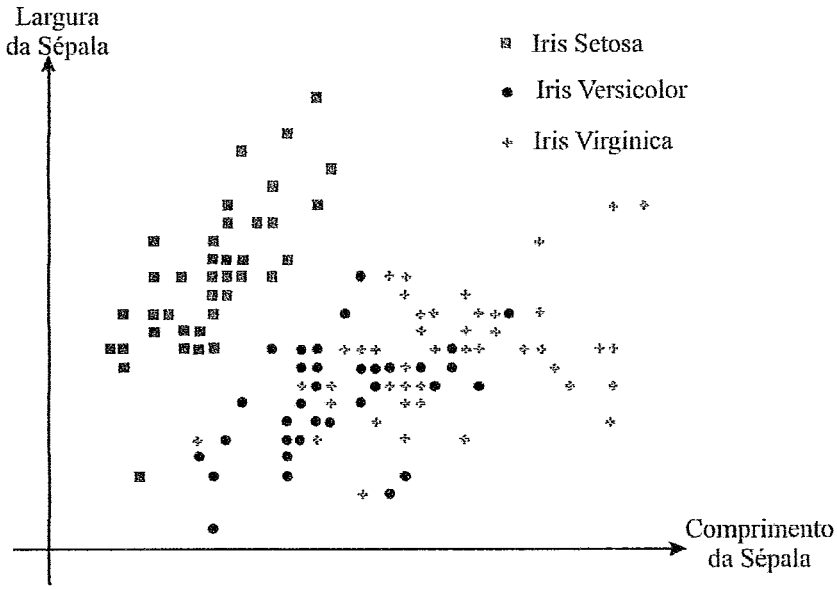


Figura 5.4 – Base de Dados Iris - características das sépalas

Como pode ser observado na Figura 5.4 as características referentes às sépalas não separam as três classes. Ao contrário, fazem com que as classes Iris Versicolor e Iris Virginica se apresentem de maneira bastante entrelaçada. Como essas características não acrescentam nenhum ganho ao processo de classificação, as mesmas não são analisadas. Apenas as características que representam melhor qualidade de informação, ou melhor separabilidade, são estudadas. Dessa forma, fica clara a importância da escolha de um conjunto de características que represente bem o conjunto estudado. Uma escolha equivocada pode conduzir a uma classificação precária ou incorreta do conjunto dos dados.

Para visualizar o problema graficamente são consideradas apenas as duas características mais importantes, ou seja, são consideradas apenas as características que possuem o maior poder de discriminação sobre a definição das classes. A redução da dimensão do problema é obtida aplicando-se técnicas de seleção de características afim de transformar o problema original definido no $\Re^4$ em um problema reduzido definido no $\Re^2$ (GUNN, 1998). Assim, cada observação do conjunto de treinamento terá apenas duas características cada uma.

A Figura 5.5 ilustra as três classes desse problema no $\Re^2$. As características que não foram eliminadas do conjunto de treinamento são referentes ao comprimento e à largura da pétala. Pode-se observar que a classe das Iris Setosa e a classe das Iris Versicolor são facilmente separadas por um hiperplano. Por outro lado, as classes Iris Versicolor e Iris Virginica são não linearmente separáveis.

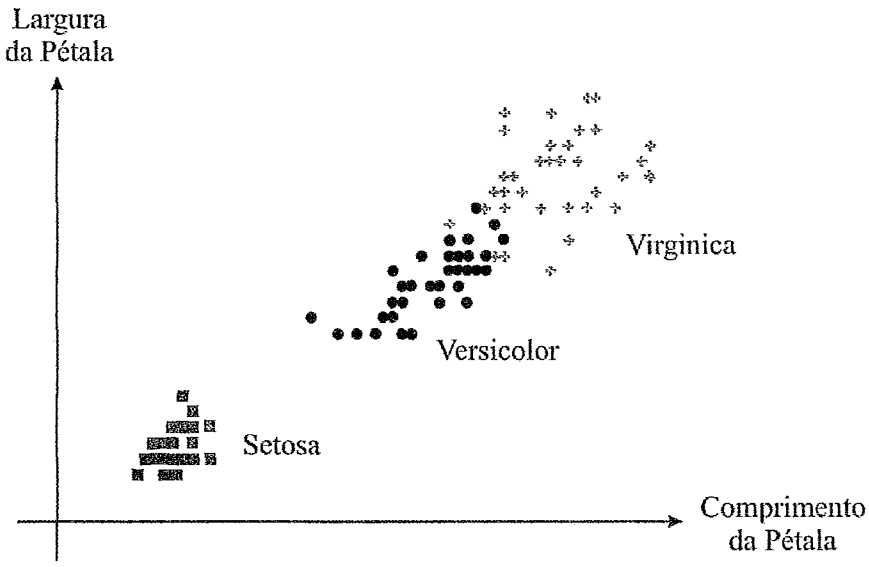


Figura 5.5 - Base de Dados Iris - características das pétalas

A validação da metodologia foi realizada solucionando o problema da Iris sobre dois enfoques. Primeiramente, construiu-se um problema teste formado por um conjunto de dados que possui apenas as classes linearmente separáveis. Posteriormente, foi construído um outro problema teste com um conjunto de treinamento que contém classes não linearmente separáveis. Assim, foi possível validar o comportamento do classificador SHSVM tanto nos problemas linearmente separáveis quanto nos não linearmente separáveis.

Os resultados obtidos pela metodologia proposta neste trabalho, SHSVM, foram comparados aos resultados obtidos pelo software SVMLight. Como o SVMLight foi desenvolvido segundo a metodologia de Vapnik foi necessário ajustar o parâmetro C para se obter resultados compatíveis.

O parâmetro C estabelece um compromisso entre o erro de treinamento e o tamanho da margem. De uma maneira mais clara, pode-se definir o parâmetro C como sendo uma penalização sobre as observações que violam a margem definida pelo classificador. Como a metodologia proposta visa unicamente maximizar a margem sem permitir violações da mesma, foi adotado $C = \infty$ quando na resolução de problemas linearmente separáveis pelo *SVMLight*. Para a resolução de problemas não linearmente separáveis utilizou-se $C \simeq 0$. Essa estratégia faz com que no caso linearmente separável não ocorra violação da margem e, em contrapartida, no caso não linearmente separável o valor da margem se torna o maior possível, pois só conta o valor da margem, já que as violações tem peso (penalização) zero.

A Tabela 5.1 apresenta os resultados obtidos para o primeiro caso descrito anteriormente, onde o conjunto de treinamento é formado apenas pelas classes linearmente separáveis.

As soluções obtidas nas fases de treinamento e de teste pela metodologia proposta SHSVM e pelo SVMLight foram iguais e corretas para a resolução do problema linearmente separável. Por outro lado, a metodologia proposta perde em termos de tempo de processamento, sendo o algoritmo SHSVM menos eficiente que o *SVMLight*.

Tabela 5.1 – Classificação entre as classes Linearmente Separáveis

Base de Dados m x n	**Algoritmo**	**Margem**	**Exatidão (%)**		**Tempo CPU (segundos)**
			Treinamento	**Teste**	
Iris 150 x 4	SHSVM	1.635107122	100	100	0.09
	SVMLight	1.635109061	100	100	0.02

Uma comparação exata entre as margens produzidas pelo SHSVM e o *SVMLight* não é possível, pela diferença de critérios. A margem do *SVMLight* será sempre um pouco maior que a margem obtida pelo SHSVM. Essa diferença decorre do SVMLight

ensejar certa tolerância quanto ao posicionamento dos pontos em relação à margem, enquanto no SHSVM não há nenhum ponto dentro da margem.

Para se ter a garantia da igualdade das duas soluções foi feita uma análise sobre os pontos suporte obtidos pelo SHSVM e pelo *SVMLight*. Concluiu-se que os pontos suporte obtidos pelos dois algoritmos são exatamente os mesmos, resultado que é indicador da validade da metodologia proposta, bem como, da perfeita igualdade das soluções.

A Figura 5.6 ilustra a solução obtida na resolução do problema Iris considerando apenas as classes linearmente separáveis mais próximas, ou seja, o conjunto de treinamento contém apenas as observações da classe Iris Setosa e da classe Iris Versicolor.

Os resultados numéricos e visuais demonstram a robustez da metodologia quando aplicada a problemas linearmente separáveis.

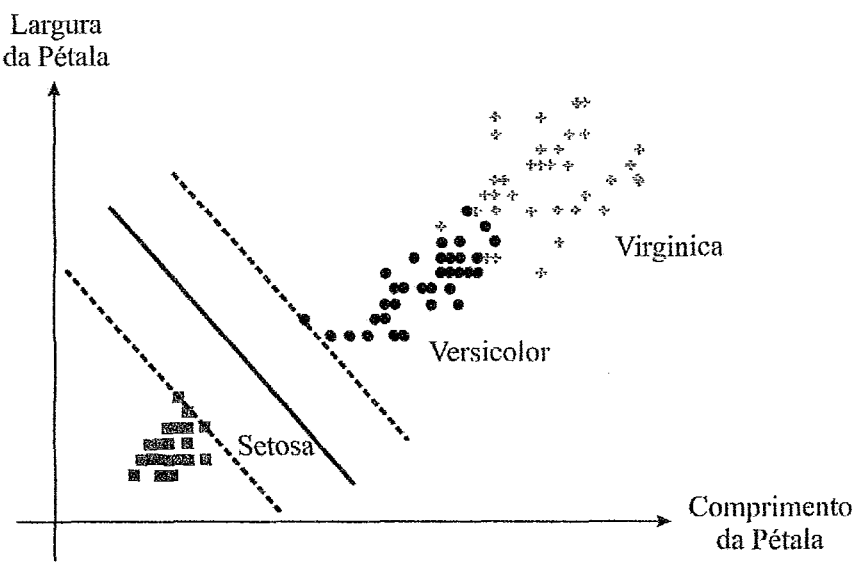


Figura 5.6 - Solução do problema Iris para classes linearmente separáveis

Os testes foram realizados utilizando-se a técnica de *leave-one-out*, que, além de ser muito utilizada para obter a exatidão dos métodos de classificação, é a técnica utilizada pelo software *SVMLight*. Essa técnica de teste consiste em retirar uma observação do conjunto de treinamento e realizar a etapa de aprendizado sobre todos os outros $m-1$ elementos do conjunto. A observação que foi retirada é classificada sobre o hiperplano separador encontrado. Esse processo é repetido para cada uma das m observações do conjunto de treinamento e, no final do mesmo, tem-se o número de

observações que foram classificadas corretamente. As tabelas deste capítulo apresentam o percentual de acertos para cada um dos problemas executados.

De outro lado, as classes Iris Versicolor e Iris Virginica são não linearmente separáveis, como pode ser observado nas figuras anteriores. Assim, para avaliar o comportamento do SHSVM aplicado a problemas não linearmente separáveis, foi construído um conjunto de treinamento formado somente por observações dessas duas classes.

O software SHSVM comportou-se de maneira estável na solução desse problema proposto buscando sempre maximizar a margem. Entretanto, há uma incompatibilidade de comparação completa dos resultados, em função da diferença entre os critérios utilizados pelos softwares SHSVM e *SVMLight*, em particular, no quesito margem. Segundo o algoritmo proposto, o valor da margem assume um valor negativo, coerente com o que deve acontecer nos casos de não separabilidade, enquanto o *SVMLight* fornece um valor extremamente alto para o tamanho da margem.

A Tabela 5.2 apresenta os resultados obtidos na resolução desse problema não linearmente separável. Observa-se que tanto na fase de treinamento quanto na fase de teste o SHSVM é bem melhor que o SVMLight, tendo um erro significativamente percentual menor nos dois casos.

Tabela 5.2 – Classificação entre as classes Não Linearmente Separáveis

Base de Dados m x n	**Algoritmo**	**Exatidão (%)**		**Tempo CPU (segundos)**
		Treinamento	**Teste**	
Iris 150 x 4	SHSVM	94	94	0.10
	SVMLight	88	90	0.03

A Figura 5.7 ilustra o bom comportamento da metodologia desenvolvida neste trabalho quando aplicada na resolução desse problema não linearmente separável. Como dito anteriormente, apenas as duas classes não linearmente separáveis, Iris Versicolor e Iris Virginica, foram utilizadas na construção do problema não linearmente separável.

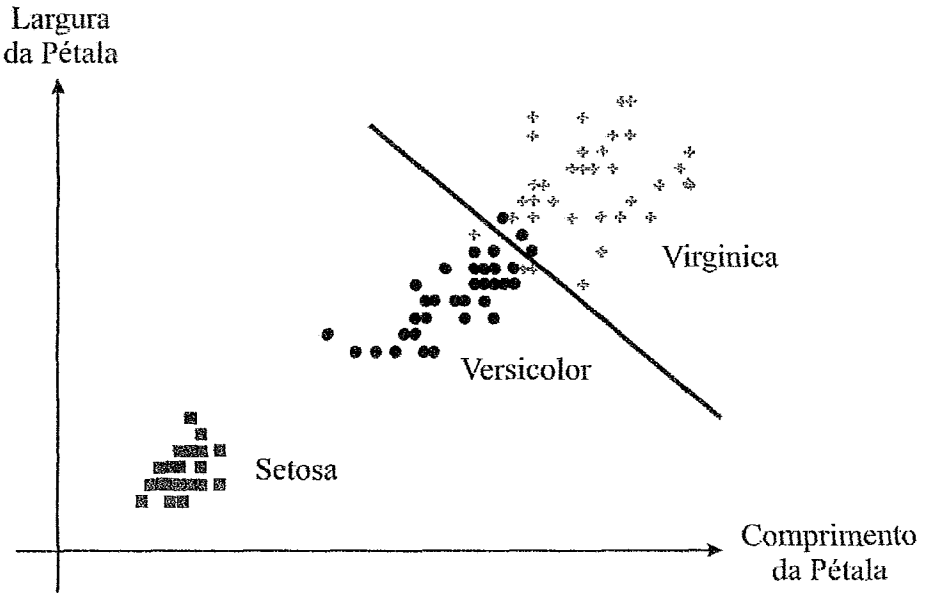


Figura 5.7 - Solução do problema Iris para classes não linearmente separáveis

Com as soluções obtidas para os dois problemas definidos anteriormente, pode-se concluir que a metodologia proposta para a resolução do problema de classificação, com enfoque SVM, se mostrou adequada nesses dois testes, quando aplicada a problemas linearmente separáveis e quando aplicada a problemas não linearmente separáveis.

5.2.1 – Desordem no Fígado (Liver-disorder)

A base de dados *Liver-Disorder* foi gerada a partir de testes sangüíneos realizados no instituto de pesquisa "*BUPA Medical Research Ltd.*" (FORSYTH, 1990). O presente conjunto de dados foi obtido através de testes sangüíneos realizados em pessoas do sexo masculino, onde foi medida a sensibilidade a afecções hepáticas devido ao consumo excessivo de álcool. Foram analisados 345 indivíduos e coletados sete tipos de informação dos mesmos.

Nesse caso-teste, não foram executados algoritmos de seleção de características. Dessa forma, o conjunto de trabalho apresenta as sete características e 345 amostras, inviabilizando uma visualização gráfica dos resultados.

Os dados apresentam a característica de serem não linearmente separáveis e, como pode ser observado na Tabela 5.3, pelos valores das colunas exatidão, o grau de separação entre as duas classes é pequeno.

Na Tabela 5.3 são apresentados os tempos de processamento e exatidão dos algoritmos, além das margens obtidas. Embora o SHSVM tenha obtido uma exatidão um pouco menor na fase de treinamento do classificador, na fase de teste essa diferença não é encontrada. Fato que valida novamente a eficiência da metodologia para dados não linearmente separáveis. Como pode ser facilmente observado, os critérios de margem para dados linearmente separáveis são diferentes nas duas implementações. No *SVMLight* o valor da norma de w é um número bastante pequeno, o que faz com que o valor da margem fique grande, enquanto no SHSVM a margem possui valor negativo, pois não é possível uma separação linear sem que dados violem a margem.

Tabela 5.3 – Resultado da classificação da base de dados *Liver Disorder*

Algoritmo	Margem	Exatidão fase de treinamento (%)	Exatidão fase de teste (%)	Tempo CPU (segundos)
SHSVM	-9,70021	57	57,97	0.31
SVMLight	20000	57,97	57,97	0.04

5.2.2 – Diabetes

Esta base de dados foi obtida a partir de estudos do instituto *"National Institute of Diabetes and Digestive and Kidney Diseases"(SIGILLITO, 1990).* Os dados foram obtidos através de estudos em mulheres com menos de 21 anos. Foram selecionadas 768 pessoas e oito características distintas. Sendo elas: o número de vezes que engravidou, a concentração de glicose em um teste oral de tolerância a glicose, a pressão arterial, a "espessura da prega cutânea", a concentração de insulina, o índice de massa corporal, o "grau" de diabetes e a idade.

Como pode ser observado na Tabela 5.4 os resultados obtidos pelos dois algoritmos são praticamente os mesmos, com exceção do tempo de processamento, onde o SHSVM se mostrou inferior em relação ao *SVMLight.* Entretanto, essa diferença não inviabiliza a utilização da metodologia proposta, pois nesses casos a preocupação maior é com a eficiência e não com a velocidade algoritmo.

A diferença encontrada no valor das margens, como dito no caso anterior, se deve ao fato dos dois algoritmos possuírem um critério diferente para o cálculo da margem.

Tabela 5.4 - Resultados de classificação da base de dados *Diabetes*

Algoritmo	Margem	Exatidão fase de treinamento (%)	Exatidão fase de teste (%)	Tempo CPU (segundos)
SHSVM	-7,0755	65	65	0.34
SVMLight	519,48	65,1	65,1	0.02

5.2.3 – Bases Aleatórias

Para analisar a eficiência da metodologia proposta, foi realizado um novo estudo comparativo entre o algoritmo desenvolvido SHSVM e o algoritmo de classificação *SVMLight*. Foram criados vários conjuntos de treinamento aleatórios com tamanhos diferentes, variando de 200 observações a 200 mil observações. Esses problemas foram criados para testar a robustez e a eficiência da nova metodologia na solução de problemas de pequeno, médio e grande porte.

Os problemas teste foram criados usando o software *NDC Data Generator*, produzido por Musicant (MUSICANT, 1998). No primeiro conjunto de teste, as observações foram geradas aleatoriamente segundo uma distribuição uniforme no intervalo]–6,8] gerando duas classes linearmente separáveis.

Como pode ser observado na Tabela 5.5 as margens encontradas pelos dois algoritmos são praticamente equivalentes. Essa pequena diferença entre os valores obtidos é justificada pela diferença no cálculo da margem. Em uma análise mais profunda, pôde-se notar que os vetores suporte para esses casos são exatamente os mesmos, o que indica a validade do algoritmo proposto SHSVM. Assim, as diferenças nos valores de margem só podem ser explicadas pela diferença de critérios de cálculo.

Os resultados exibidos nesta tabela também mostram um domínio inequívoco e destacado do *SVMLight* no quesito tempo. A relação entre os tempos dos dois métodos, SHSVM e *SVMLight*, assumem os valores (0.0625, 4.33, 11.18, 21.36), ou seja, são crescentes com o tamanho do problema.

Pela análise dos detalhes de implementação do *SVMLight* (JOACHIMS, 1999) pode-se concluir que essa diferença em relação ao tempo de processamento, está diretamente relacionada às técnicas de redução do conjunto de treinamento. Essas demonstram, assim, sua grande eficiência. Ademais, essa eficiência é crescente com o temanho do problema, como seria de se esperar, pois o número de pontos candidato a suporte é percentualmente menos em relação ao número total de pontos.

Tabela 5.5 - Comparação dos resultados para bases aleatórias

Base de Dados m x n	Algoritmo	Margem	Exatidão fase de teste (%)	Tempo CPU (segundos)
Base 200	SHSVM	15.65636759185	100	0.03
200 x 10	*SVMLight*	15.65680288085	100	0.48
Base 2000	SHSVM	3.106290251622	100	1.30
2000 x 10	*SVMLight*	3.106361829025	100	0.03
Base 20000	SHSVM	3.101742156230	100	55.90
20000 x 10	*SVMLight*	3.102843756302	100	0.50
Base 200000	SHSVM	0.025655465151	100	193.77
200000 x 10	*SVMLight*	0.025922450397	100	9.07

Atendo-se ao desempenho do SHSVM, vê-se primeiramente que para o problema de menor dimensão, Base 200, o seu tempo de processamento foi 16 vezes mais rápido, exatamente onde os ganhos de redução do conjunto de treinamento do *SVMLight* são proporcionalmente menores. Essa constatação permite vislumbrar ganhos de eficiência no método SHSVM quando da implementação de técnicas de redução análogas às do *SVMLight,* tornando-o competitivo no quesito tempo e mantendo a mesma eficácia.

Finalmente, deve ser explicitado que nos experimentos computacionais descritos neste capítulo foi realizado um trabalho de harmonização dos sete parâmetros definidos anteriormente. Embora os resultados mostrem que a combinação de valores para os parâmetros tenha sido adequada, nem todas as opções de melhoria foram exauridas, ou seja, uma harmonização melhor ainda pode ser obtida.

Capítulo 6 – Conclusões

Neste trabalho, foi descrito o processo de Mineração de Dados com enfoque em uma de suas fases, a classificação. Como a qualidade do processo de MD como um todo depende essencialmente da qualidade do classificador, uma ferramenta de classificação eficaz se torna fundamental na obtenção de bons resultados. Assim, foi utilizada como base de desenvolvimento a metodologia SVM, que apresenta resultados consagradamente satisfatórios na resolução de problemas de classificação, conforme consignado na literatura. Com o objetivo de resolver um modelo matemático associado à metodologia SVM, foi apresentada uma nova proposta metodológica denominada SHSVM. Essa proposta se caracteriza por ser constituída pela articulação das técnicas de Suavização Hiperbólica e Penalização Hiperbólica. O que permite a resolução do problema SVM na sua forma primal.

A partir dos resultados obtidos, pode-se concluir que o método apresentou um bom desempenho, oferecendo os atributos de robustez e eficiência. Sendo assim, constitui-se em uma nova proposta na solução de problemas de classificação que surgem em Mineração de Dados.

A harmonização do conjunto de parâmetros mostrou-se adequada, entretanto nem todas as alternativas foram exauridas. Assim, provavelmente, pode ser obtida uma melhor combinação de valores para esses parâmetros, de forma que os resultados, no quesito tempo, melhorem ainda mais.

O uso de funções Kernel para a geração de superfícies separadoras não lineares para problemas não linearmente separáveis também pode ser aplicado a esta metodologia, ampliando e melhorando a aplicabilidade do método proposto.

Além da não-linearidade das superfícies, um outro aprimoramento seria a permissão à violação da margem, com adoção de um esquema de penalização. Atualmente não é permitido que um ponto viole a margem. Com isso, conjuntos não linearmente separáveis resultam em uma margem negativa. Entretanto, a violação de um pequeno conjunto de pontos pode não significar inseparabilidade e sim um dado com ruído, por exemplo. Com a permissão da violação, os dados seriam da mesma forma classificados e o resultado do valor da margem apresentaria valor positivo. O dado violado seria penalizado, mas não alteraria o valor da margem de forma a torná-la

negativa. Essa abordagem permitiria a resolução de uma alternativa do SVM que poderíamos denominar SVM de margem *soft*.

Em suma, a análise dos resultados do método comprovam a eficiência e a robustez da metodologia, e, por via de conseqüência, sua aplicabilidade, podendo a mesma ser utilizada nos mais diversos estudos de classificação.

6.1 – Propostas para trabalhos futuros

Como extensões naturais ao presente trabalho poderia ser relacionada uma série de desdobramentos.

Seria indicada a criação de um software que unisse à metodologia apresentada, técnicas de seleção de características e de tratamento de *missing values*, para a solução de problemas mais abrangentes de Mineração de Dados. Com isso, seria possível a utilização do mesmo em um processo mais amplo de MD.

Para uma maior aplicação da metodologia de classificação discutida pode-se cogitar o emprego de uma abordagem mais ampla do problema SVM. Com isso, a proposta poderia ser utilizada na resolução de problemas mais complexos, como os problemas não linearmente separáveis.

Nesse sentido, uma boa alternativa seria a utilização de funções *kernel* na formulação matemática do problema SVM, a fim de obter separadores não lineares na resolução de problemas de classificação. O uso de *kernel* possibilita que os dados do conjunto de treinamento sejam separados por uma superfície, não linear, que poderá ter um maior poder de generalização que uma superfície linear.

Uma outra alternativa seria incluir critérios de poda para tentar reduzir ainda mais o tempo de processamento.

Outras melhorias nos tempos de processamento podem ser obtidas com a adoção de esquemas de redução de observações, análogos aos implementados no *SVMLight*.

Apêndice A. Medidas de proximidade

As métricas mais utilizadas para medir a proximidade entre dois pontos são a distância Euclidiana, a distância de Mahalanobis e a distância de Manhattan. A seguir são descritas cada uma dessas distâncias.

A.1 - Distância Euclidiana

A distância Euclidiana é a métrica mais utilizada e é calculada como a distância em linha reta entre os dois pontos. Como mostrado na Figura 1.6.1.

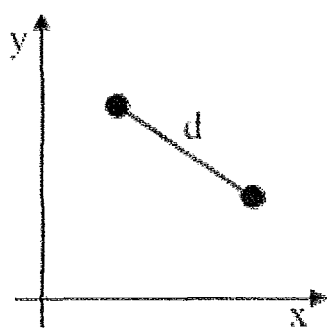


Figura 1.6.1 - Distância Euclidiana

É calculada através da seguinte equação:

$$d(x,y) = \|x-y\| = \sqrt{(x-y)^t (x-y)}$$

A.2 - Distância de Manhattan ou 'city-block'

A distância de Manhattan é uma métrica de cálculo simples, por isso é bastante utilizada em implementações em tempo real. É uma simplificação da distância euclidiana. É calculada pela soma dos catetos, ao contrário da euclidiana que é dada pelo valor da hipotenusa.

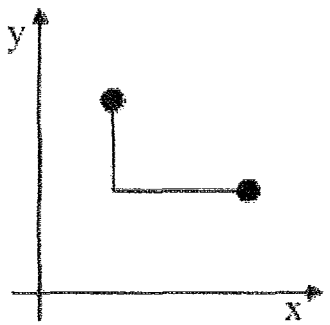


Figura 1.6.2 - Distância de Manhattan

É calculada através da seguinte equação:

$$d(x,y)=\sum_{i=1}^{n}|x_i-y_i|$$

onde x_i e y_i, são, respectivamente, o valor do i-ésimo atributo para as observações x e y.

A.3 - Distância de Mahalanobis

Esta métrica possui um cálculo mais complexo em relação às outras citadas acima. Ela considera que as superfícies de cada classe são elipsóides centrados na média. No caso especial em que a covariância é zero e a variância é a mesma para todas as variáveis, as superfícies são esferas, e a distância de Mahalanobis se torna equivalente à distância Euclidiana. Pode ser bastante difícil se determinar precisamente as matrizes de covariância, e o custo computacional cresce bastante com o número de variáveis envolvidas, por isso, em geral, prefere-se usar a distância Euclidiana.

É calculada pela seguinte equação:

$$d\left(x^T,y^T\right)=\sqrt{(x-y)S^{-1}(x-y)^t}$$

onde S é a matriz de covariância das observações que pode ser definida como:

$$S=matriz\left(s_{kj}\right)=\frac{1}{n}\sum_{i=1}^{n}\left(x_{ik}-\overline{x_k}\right)\left(x_{ij}-\overline{x_j}\right)$$

onde $k, j = 1..p$, p é o número de atributos, x_{ik} e x_{ij} são, respectivamente, os valores da observação da linha i e a coluna k e j, e $\overline{x_k}$ e $\overline{x_j}$ são, respectivamente, as médias correspondentes das observações ao longo das colunas k e j.

ASUNCION, A., NEWMAN, D.J., 2007, "UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]", *University of California, Irvine, School of Information and Computer Sciences*, 12/01/2008.

BENNETT, K.P., 1992, "Decision Tree Construction via Linear Programming ". In: *Proceedings of the 4th Midwest Artificial Intelligence and Cognitive Science Society Conference*, pp. 97-101, Utica, Illinois.

BENNETT, K.P., O.L.MANGASARIAN, 1992, "Neural network training via linear programming". In: PARDALOS, P.M. (eds), *Advances in Optimization and Parallel Computing*, Amsterdã.

BRADLEY, P.S., FAYYAD, U.M., MANGASARIAN, O.L., 1998, "Mathematical Programming for Data Mining: Formulations and Challenges", *Journal on Computing*, v. 11 (1999), pp. 217-238.

BREIMAN, L., FRIEDMAN, J.H., OLSHEN, R.A., *et al.*, 1983, *Classification and Regression Trees*, Belmont, CA, Wadsworth Publishing Company.

BRITO, J.A.M., 2004, *Suavização Hiperbólica Aplicada no Problema de Localização de Estações de Rádio Base*, Tese D. Sc., COPPE, UFRJ, Rio de Janeiro, RJ, Brasil.

BUNTINE, W.L., 1996, "Graphical models for discovering knowledge", *Advances in Knowledge Discovery and Data Mining*, Menlo Park, CA, AAAI Press.

BURGES, C.J.C., 1998, "A Tutorial on Support Vector Machine for Pattern Recognition", *Data Mining and Knowledge Discovery*, v. 2, pp. 121-167.

CHAVES, A.M.V., 1987, *Resolução de Problemas Minmax Via Suavizações*, Dissertação de M. Sc., COPPE, UFRJ, Rio de Janeiro, RJ.

CHAVES, A.M.V., XAVIER, A.E., 1998, *Problemas Minimax: Uma Alternativa de Resolução via suavização*, Relatório Técnico, COPPE, UFRJ, Rio de Janeiro, RJ, Brasil.

CORTES, C., VAPNIK, V., 1995, "Support Vector Networks", *Machine Learning*, v. 20, pp. 273-297.

DIB, K.R., 1994, *Utilização de Função de Penalização Hiperbólica na Suavização e Otimização de um Modelo Chuva-Vazão: Modelo SWMS*, Tese de D.Sc., COPPE, UFRJ, Rio de Janeiro.

DIETTERICH, T.G., 2000, "Ensemble methods in machine learning", *Lecture Notes in Computer Science*, v. 1857, pp. 1-15.

EVERITT, B.S., DUNN, G., 1983, *Advanced Methods of Data Exploration and Modeling*, London, Heinemann Educational Books.

FÉLIX, L.C.M., REZENDE, S.O., MONARD, M.C., *et al.*, 2000, "Transforming a regression problem into a classification problem using hybrid descretization", *Computatión y Sistemas*, v. 4, pp. 44-52.

FISHER, R.A., 1935, *The logic of inductive inference.*, J. R. Stat. Soc.

FORSYTH, R.S., 1990, "BUPA liver disorders", *BUPA Medical Research Ltd.*

GOLSCHIMDT, R., PASSOS, E., 2005, *Data Mining - Um Guia Pratico*, Editora Campus / Elsevier.

GOLSCHIMIDT, R., PASSOS, E., 2005, *Data Mining - Um Guia Pratico*, Editora Campus / Elsevier.

GUNN, S.R., 1998, *Support Vector Machines for Classification and Regression*, Department of Electronics and Computer Science, University of Southampton.

HANSEN, P., JAUMARD, B., 1997, "Cluster Analysis and Mathematical Programming", *Mathematical Programming*, n. 79, pp. 191-215.

HECKERMAN, D., 1997, "Bayesian networks for data mining", *Data Mining and Knowledge Discovery.*

INMON, W.H., 1996, "The Data Warehouse and Data Mining", *Communications of ACM*, v. 39, n. 11, pp. 49-50.

J. HERTZ, A. KROGH, PALMER, R.G., 1991, *Introduction to the Theory of Neural Computation*, Redwood, CA.

JAIN, A.K., DUIN, R.P.W., MAO, J., 2000, "Statistical Pattern Recognition: A review", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 22(1), pp. 4-37.

JOACHIMS, T., 1998, *Making large-Scale SVM Learning Practical.*, Computer Science Department of the University of Dortmund, 24.

JOACHIMS, T., 1999, *Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf and C. Burges and A. Smola ed., MIT Press.

KAUFMAN, L., ROUSSWEUW, P., 1990, *Finding Groups in Data: An Introductory to Cluster Analysis*, New York.

KEARNS, M.J., VAZIRANI, U.V., 1994, "An introduction to computational learning theory", *Ellis Horwood.*

LAHENBRUCH, P.A., MICKEY, R.M., 1968, "Estimation of errors rates in discriminant analysis", v. 10, pp. 1-11.

LAVRAC, N., FLACH, P., ZUPAN, R., 1999, "Rule evaluation measures: A unifying view". In: *Proceedings of the Ninth Internationa Workshop on Inductive Logic Programming (ILP-99)*, v. 1634, pp. 74-185.

LIU, B., HSU, W., 1996, "Post-analysis of learned rules", *AAAI.*

MANGASARIAN, O.L., 1965, "Linear and nonlinear separation of patterns by linear programming", *Operations Research*, v. 13, pp. 444 - 452.

MANGASARIAN, O.L., MUSICANT, D.R., 1999, *Data Discrimination Via Nonlinear Generalized Support Vector Machines*, Computer Sciences Department, University of Wisconsin, 96-05.

MCCLELLAND, J.L., RUMMELHART, D.E., 1987, *Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises*, Cambridge, Massachusetts, MIT Press.

MINOUX, M., 1986, "Mathematical Programming: Theory and Algorithms", *John Wiley and Sons, Chichester.*

MOTA, F.C., BHAYA, A., KASKUREWICZ, E., 1992, *"Robust Stabilization of Time-Varying Discrete Internal Systems", Proceedings of Congress of Decision and Control (CDC-92)*, Tucson, Arizona.

MURPHY, P.M., AHA, D.W., 1992, "UCI Repository of Machine Learning Datasets", www.ics.uci.edu/mlearn/MLRepository.html.

MUSICANT, D.R., 1998, "NDC: Normally Distributed Clustered Datasets", *Computer Sciences Department, University of Wisconsin, Madison*, www.cs.wisc.edu/dmi/svm/ndc/.

PLATT, J.C., 1998, *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines.*, Technical Report MSR-TR-98-14.

QUINLAN, J.R., 1993, *Programs for Machine Learning*, San Mateo, CA, Morgan-Kaufman.

R. O. DUDA, P.E.H., 1973, *Pattern Classification and Scene Analysis*, New York, John Wiley and Sons.

REZENDE, S.O., PUGLIESI, J.B., MELANDA, E.A., *et al.*, 2003, "Mineração de Dados". In: MANOLE (eds), *Sistemas Inteligentes: Fundamentos e Aplicações*, Barueri, SP.

SANTOS, A.B.A., 1997, *Problemas de Programação Não-Diferenciável: Uma Metodologia de Suavização*, Dissertação de M. Sc., COPPE, UFRJ, Rio de Janeiro, RJ.

SIGILLITO, V., 1990, "Pima Indians Diabetes Database", *RMI Group Leader.*

SILVA, L.P., XAVIER, A.E., CANEDO, M.P., 1990, *Calibração Automática de Modelos Chuva-Vazão: Um método Assintótico*, COPPE, Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brasil.

STONE, M., 1974, "Cross validation choice and assesment of statistical predictions", *Journal of the Royal Statistical Society*, v. 36, pp. 111-147.

THEODORIDIS, S., KOUTROUMBAS, K., 2006, *Pattern Recognition*, 3ª ed., USA, Academic Press.

VAPNIK, V.N., 1995, *The nature of statistical learning theory*, Springer-Verlag New York, Inc.

WAHBA, G., 1990, "Spline Models for Observational Data", SIAM, Philadelphia.

WAHBA, G., WANG, Y., GU, C., *et al.*, "Structured machine learning for 'soft' classification with smoothing spline anova and stacked tuning, testing and evaluation", *Advances in Neural Information Processing Systems*, San Mateo, CA, Morgan Kaufmann.

WEISS, S.M., INDURKHYA, N., 1998, *Predictive Data Mining: A Practical Guide*, San Francisco, CA, Morgan Kaufmann Publishers, Inc.

WIKIPÉDIA, 2008, "Mineração de dados". In: http://pt.wikipedia.org/wiki/Minera%C3%A7%C3%A3o_de_dados, accessed in 07/01/2008.

XAVIER, A.E., 1982, *Penalização Hiperbólica: Um Novo Método para Resolução de Problemas de Otimização*, COPPE, UFRJ, Rio de Janeiro.

XAVIER, A.E., 1993, *Solução de Problemas de Programação Não-Diferenciáveis via Suavização*, Relatório Técnico ES-290/93, PESC/COPPE, UFRJ, Rio de Janeiro.

XAVIER, A.E., 2000, "Optimum Covering of Plane Domains by Circles". In: *17th International Symposium on Mathematical Programming*, Atlanta, USA.

XAVIER, A.E., 2005, *The Hyperbolic Smoothing Clustering Method*, Relatório Técnico 674/05, PESC / COPPE, UFRJ.

XAVIER, A.E., CURVELO, P., STRÖELE, V., *et al.*, 2006, "The Hyperbolic Smoothing Approach for Solving the Support Vector Machine Problem". In: *19 th International Symposium on Mathematical Programming*, Rio de Janeiro, RJ.

XAVIER, A.E., OLIVEIRA, A.A.F., 2003, "Optimum Order p Covering of Plane Domains by Circles Via Hyperbolic Smoothing Method". In: *International Symposium on Mathematical Programming*, Kopenhagen.