SELEÇÃO DE CARACTERÍSTICAS COM BUSCA ORDENADA E CLASSIFICADORES DE LARGA MARGEM

Saulo Moraes Villela

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

Orientadores: Adilson Elias Xavier
Raul Fonseca Neto

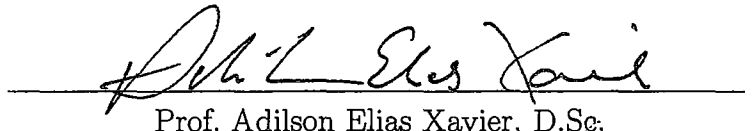
Rio de Janeiro
Dezembro de 2011

SELEÇÃO DE CARACTERÍSTICAS COM BUSCA ORDENADA E CLASSIFICADORES DE LARGA MARGEM

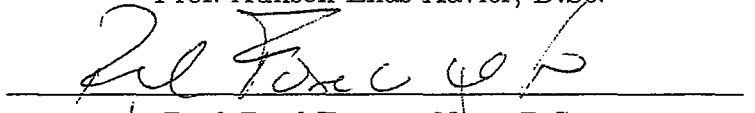
Saulo Moraes Villela

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

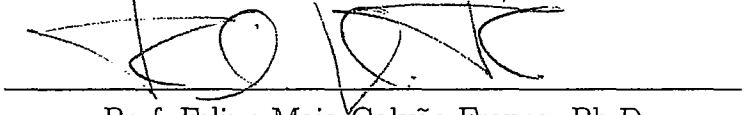
Examinada por:



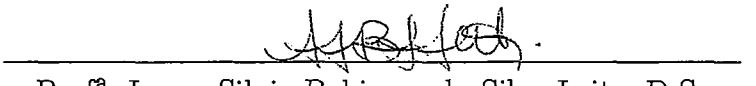
Prof. Adilson Elias Xavier, D.Sc.



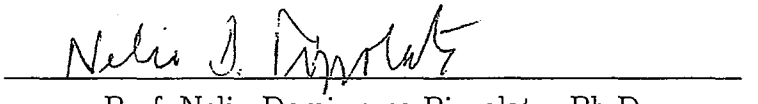
Prof. Raul Fonseca Neto, D.Sc.



Prof. Felipe Maia Galvão França, Ph.D.



Prof[a]. Laura Silvia Bahiense da Silva Leite, D.Sc.



Prof. Nelio Domingues Pizzolato, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
DEZEMBRO DE 2011

Moraes Villela, Saulo

Seleção de Características com Busca Ordenada e Classificadores de Larga Margem/Saulo Moraes Villela. – Rio de Janeiro: UFRJ/COPPE, 2011.

XIV, 92 p.: il.; 29, 7cm.

Orientadores: Adilson Elias Xavier

Raul Fonseca Neto

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2011.

Referências Bibliográficas: p. 85 – 92.

1. Seleção de Características. 2. Classificador de Larga Margem. 3. Busca Ordenada. 4. Margem Projetada. I. Elias Xavier, Adilson *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

"Vida sã é aquela que se inspira
no amor e se guia pelo
conhecimento."
(Bertrand Russel)

Ao meu avô Herbert pelos
ensinamentos e lições

Agradecimentos

Agradeço, primeiramente, a Deus a bênção da vida. Tê-Lo como guia é fundamental para se obter o sucesso.

Agradeço à minha mãe por ter incentivado-me, sempre, a buscar meus objetivos, privando-se de alguns de seus sonhos para que eu e minha irmã pudéssemos realizar os nossos.

Agradeço ao meu pai a cumplicidade e convivência.

Agradeço à minha irmã Tatyana por acreditar em mim e apoiar-me, incondicionalmente, em minhas decisões.

Agradeço à minha avó Teresinha o incentivo e a presença constantes.

Agradeço à Cecília o companheirismo na reta final.

Agradeço à professora Inês de Castro Dutra por ter me aceitado no Doutorado.

Agradeço ao meu orientador Adilson Elias Xavier por me acolher, quando eu estava "órfão", e ter confiado no meu trabalho.

Agradeço ao meu orientador Raul Fonseca Neto que, desde o tempo da graduação, incentivou-me na pesquisa, e presenteou-me com esse belíssimo tema. Meu agradecimento também pela primeira oportunidade de docência.

Agradeço, também, aos meus orientadores, além dos ensinamentos, a amizade nesta longa caminhada.

Agradeço ao amigo Saul de Castro Leite, que presenteou-me, também, com o tema da minha pesquisa, o apoio incansável a mim dispensado quando do desenvolvimento do projeto.

Agradeço aos professores Felipe Maia Galvão França, Laura Silvia Bahiense da Silva Leite e Nelio Domingues Pizzolato a participação na banca examinadora. Ao professor Felipe, além do professor Carlos Cristiano Hasenclever Borges, agradeço, também, a participação na banca de qualificação.

Agradeço aos professores e funcionários do PESC, em especial à Solange, pelo suporte necessário.

Agradeço aos amigos do DCC, Granbery e FAGOC as oportunidades e trocas de experiências.

Agradeço a todos os meus alunos, com os quais, durante os últimos anos, tenho aprendido muito mais do que ensinado.

Agradeço à minha tia Marlene que me acolheu quando precisei.

Agradeço a todos os meus familiares e amigos, presentes em todas as ocasiões, os cuidados, a força, o carinho e a atenção que tornaram possível este momento.

A todas as outras pessoas que colaboraram direta ou indiretamente nessa minha conquista, agradeço sinceramente.

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

SELEÇÃO DE CARACTERÍSTICAS COM BUSCA ORDENADA E CLASSIFICADORES DE LARGA MARGEM

Saulo Moraes Villela

Dezembro/2011

Orientadores: Adilson Elias Xavier
Raul Fonseca Neto

Programa: Engenharia de Sistemas e Computação

Classificadores de larga margem, como as máquinas de vetores suporte, têm sido amplamente utilizados com bastante eficiência em problemas que possuem uma grande quantidade de variáveis ou características. Nesse sentido, torna-se importante o desenvolvimento de novas estratégias para a solução desse tipo de problema que se utilizem de critérios de seleção associados a esse tipo de classificador. Nesse trabalho propõe-se um novo método para a seleção de subconjuntos de variáveis que utiliza um processo de busca ordenada, também conhecido como *best-first*, para exploração do espaço de possíveis candidatos. O algoritmo, denominado AOS, utiliza como medida de avaliação os valores de margem calculados a partir da utilização de um classificador de larga margem. Esse classificador, de grande eficiência computacional, permite grande flexibilidade e rapidez na obtenção dos valores de margem possibilitando a solução de problemas de tamanho razoável, com centenas de características, sem que ocorra explosão combinatória. O algoritmo foi testado em vários problemas da literatura e seus resultados comparados à outras técnicas de seleção de subconjuntos. Uma importante contribuição teórica do trabalho se refere ao desenvolvimento do conceito de margem projetada. A utilização desse valor, computado como a projeção da margem real de um espaço em cada subespaço de dimensão inferior permitiu maior eficiência e rapidez na solução dos problemas de classificação e, portanto, no processo de busca como um todo.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

FEATURE SELECTION WITH ORDERED SEARCH AND LARGE MARGIN CLASSIFIERS

Saulo Moraes Villela

December/2011

Advisors: Adilson Elias Xavier
Raul Fonseca Neto

Department: Systems Engineering and Computer Science

Large margin classifiers, as support vector machines, have been widely used quite effectively on high dimensional problems. In this sense, it is important to develop new feature selection strategies that are associated with this type of classifier. In this work, we propose a new method for feature selection based on an ordered search process, also known as best-first, to explore the space of possible candidates. The algorithm, called AOS, uses as a search measure the margin values calculated using a large margin classifier. This highly efficient classifier allows great flexibility and speed in obtaining the margin values, enabling the solution of problems of reasonable size, with hundreds of features, and avoiding combinatorial explosion. The algorithm was tested on several problems from the literature and the results were compared to other methods. An important theoretical contribution of the paper refers to the concept of the projected margin. This value, computed as the projection of the maximal margin vector on a lower dimensional subspace, is used as an upper bound to the actual maximal margin. This enables greater efficiency and speed in solving problems of classification and, therefore, in the search process as a whole.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

AOS	Admissible Ordered Search, p. 57
AROM	Approximation of the Zero-Norm Minimization, p. 54
ERM	Empirical Risk Minimization, p. 6
GMFP	Geometric Margin Fixed Perceptron, p. 14
IMA	Incremental Margin Algorithm, p. 31
LMS	Least Mean Square, p. 9
NSC	Nearest Srhunken Centroid, p. 54
RFE	Recursive Feature Elimination, p. 56
SMO	Sequential Minimal Optimization, p. 39
SRM	Structural Risk Minimization, p. 6
SVM	Support Vector Machines, p. 22

Capítulo 1

Introdução

There are many branches of learning theory that have not yet been analyzed and that are important both for understanding the phenomenon of learning and for practical applications. They are waiting for their researchers.

Vladimir Vapnik

1.1 Contexto e Motivação

As pessoas vivem em um mundo discriminatório. Onde quer que estejam, estão sempre observando as características ou atributos do que pode ser visto, tocado, escutado, cheirado e/ou degustado. Isso é feito com um único propósito: discriminar.

Para se qualificar, dentre outras coisas, é necessária uma observação atenta do objeto a ser rotulado. Após a percepção das características do objeto, este é, baseando-se em classificações anteriores, discriminado. O conhecimento é construído com base em informações já obtidas e assimiladas como verdade. Considere, por exemplo, o fato de se visualizar uma pessoa que não é conhecida. O cérebro, rapidamente, baseado em verdades já estabelecidas, é capaz de julgar se o novo exemplar é bonito ou feio, gordo ou magro, alto ou baixo e assim por diante.

As características desempenham o papel fundamental no processo de julgamento, já que são elas que verdadeiramente determinam a que classe o exemplar pertence. Porém, nem toda classificação é realizada em relação a critérios subjetivos, como as citadas anteriormente. Cientistas estão interessados em discriminar os padrões entre classes como venenoso ou comestível, cancerígeno ou normal, verdadeiro ou falso, dentre outros. Desta forma, o controle de erro de classificação deve ser levado

em consideração como parâmetro indicativo da qualidade do processo empregado em questão.

Um erro de classificação ocorre quando equivocadamente atribui-se um exemplar pertencente a uma determinada classe à outra. Por exemplo, considere a existência de duas classes X e Y. Se um dado exemplar pertencer à classe X e, baseado na análise de seus atributos, for atribuído à classe Y, significa que ocorreu um erro de classificação. O erro ocorre, da mesma forma, se o exemplar pertencer à classe Y e erradamente for classificado como pertencente à classe X.

Não há sentido em se avaliar erros de classificação baseados em critérios subjetivos. Porém, em problemas reais é extremamente importante o conhecimento sobre a quantidade de erro produzido. De maneira geral, quanto menos erros são cometidos, mais poderoso e preciso é o processo de classificação desenvolvido. Sem dúvida, o cérebro humano não é o único sistema capaz de realizar classificações. Com o uso da computação foi possível a construção de máquinas capazes de classificar padrões a elas apresentados. Tais máquinas receberam o nome de classificadores.

Os classificadores recebem, como entrada, as características do exemplar a ser rotulado e retornam, como saída, a classe à qual o objeto pertence. Para a viabilização deste processo, entretanto, todas as características, bem como as classes existentes, são codificadas em formato numérico para poderem ser interpretadas pelo classificador.

O classificador, independente de qual seja, baseia-se na avaliação de todas as características do exemplar a fim de proceder a classificação. Não se pode dizer que o mesmo ocorra no cérebro humano, já que este é capaz de selecionar rapidamente certos atributos que melhor classificarão o dado visualizado.

Considere, por exemplo, a tarefa de se discriminar entre sexos uma determinada pessoa. O problema é de solução trivial até mesmo para uma criança. Isso ocorre porque ela selecionará certas características que melhor discriminam as pessoas entre as classes (no caso masculino e feminino) e ignorará diversas outras. Agindo dessa maneira ela será capaz de classificar com velocidade, e sem erro, os exemplos a ela apresentados.

De maneira oposta, os classificadores irão se basear em todas as características dos padrões apresentados, para só então fornecer uma resposta. Portanto, o processo de classificação se torna lento e o número de erros relacionados pode crescer na medida em que a quantidade de atributos para representar os dados aumenta.

Esse fato é de fácil compreensão quando se imagina que um classificador, para efetuar a discriminação citada, utiliza todos os atributos relativos a uma pessoa, como cor dos olhos, cor dos cabelos, cor da pele, altura, peso etc.

O controle da quantidade de erros de classificação é, portanto, essencial em problemas reais. Com isso, uma das formas de se melhorar o desempenho de um

classificador, baseia-se em construir meios de selecionar as características que mais influenciam no processo de classificação ou discriminação.

1.2 Objetivos

O objetivo principal do processo de seleção de características é a eliminação de variáveis irrelevantes com o intuito de produzir subconjuntos de variáveis relevantes que sejam capazes de generalizarem melhor para um dado problema de classificação. Também, podem-se destacar como importantes questões relativas ao requerimento de tempo de computação, descoberta de variáveis que têm maior poder discriminante, como em análise de genes, bem como uma melhor visualização e interpretação dos resultados. Nesse sentido, considera-se nesse trabalho a investigação da eficiência da utilização das máquinas de vetores suporte associadas a um processo ordenado de seleção de candidatos na obtenção dos subconjuntos com maior poder de generalização. Adota-se, para tanto, uma estratégia de solução do tipo reversa na qual as variáveis com menor poder de discriminação são retiradas do problema. Ao contrário dos algoritmos míopes, que retiram uma variável por vez de forma irrevogável definindo uma sequência de subconjuntos aninhados, emprega-se um processo de busca ordenada que gera uma árvore de possibilidades e permite uma maior exploração da interdependência entre o conjunto de variáveis do problema.

Como forma de evitar a explosão combinatória decorrente do número exponencial de possibilidades, utiliza-se de duas estratégias que permitem controlar o processo de busca. Primeiramente, a quantidade de variáveis é reduzida até um tamanho gerenciável com a utilização de uma técnica míope, como a eliminação de variáveis sugerida pelo algoritmo RFE ou por métodos de filtragem que se baseiam no estabelecimento de um *ranking* de variáveis segundo medidas obtidas por critérios estatísticos ou de informação. Em segundo lugar, limita-se o fator de ramificação com a retirada de no máximo três possíveis variáveis a cada nível da árvore de busca, além de se eliminar estados que não tendem a gerar boas soluções. Nesse trabalho, emprega-se com esse objetivo a utilização de um classificador de larga margem que minimiza em seu treinamento a norma L_1 do vetor w no sentido de obter soluções mais esparsas.

Com isso, o objetivo do trabalho é a introdução de um algoritmo de seleção de características, AOS, que utiliza critérios e medidas provenientes de classificadores de larga margem. Para isso, é apresentado todo o embasamento do problema. São demonstrados, também, alguns algoritmos existentes na literatura. O algoritmo apresenta um processo eficaz para selecionar as características que melhor representam os dados, diminuindo, assim, o espaço representativo do problema.

1.3 Organização

Além da introdução, o trabalho está organizado em mais cinco capítulos. No Capítulo 2, é apresentado todo o embasamento sobre classificadores de larga margem. São apresentados os algoritmos IMA, com suas formulações arbitrárias, e SMO. O Capítulo 3 aborda o problema de seleção de características, com alguns dos principais algoritmos utilizados para sua solução, como o Golub, RFE, NSC e o próprio IMA, na sua formulação L_∞. O Capítulo 4 dá destaque ao algoritmo AOS, algoritmo fruto desse trabalho, descrevendo os critério estabelecidos para a retirada de variáveis e medidas de avaliação, como exemplo, as margens real e projetada. As bases de dados utilizadas e os experimentos realizados para parametrizar o algoritmo e para compará-lo com os demais, juntamente com os resultados obtidos, são exibidos no Capítulo 5. Por fim, no Capítulo 6, são apresentadas as conclusões do trabalho e algumas considerações finais, bem como sugestões de trabalhos futuros.

Capítulo 2

Classificadores de Larga Margem

Geometry is illuminating; probability theory is powerful.

Pál Ruján

Um dos principais problemas relacionados à teoria do aprendizado de máquinas consiste na definição ou aprendizado de uma função, ou hipótese, capaz de discriminar um conjunto de pontos, pertencentes a duas classes distintas, em um espaço real R^d, de dimensão d.

ROSENBLATT [1] apresentou um algoritmo extremamente simples e eficaz para o treinamento de uma unidade lógica *threshold*, capaz de encontrar uma solução na forma de um classificador linear ou hiperplano. BLOCK [2] e NOVIKOFF [3] mostraram que este algoritmo, denominado Perceptron, convergia em um número finito de iterações, caso o conjunto de pontos fosse linearmente separável. Tal algoritmo está diretamente relacionado à técnica matemática de relaxação desenvolvida para a determinação de uma solução viável para um sistema de inequações, AGMON [4], MOTZKIN e SCHOENBERG [5]. Entretanto, como a maior parte dos problemas relacionados ao aprendizado de hipóteses ou aprendizado de conceitos é de natureza não linearmente separável, MINSKY e PAPERT [6] em sua obra Perceptron, lançaram uma série de dúvidas sobre a capacidade de aprendizado desses classificadores lineares.

Anteriormente, AIZERMAN *et al.* [7] mostraram a possibilidade de utilização de funções *kernel* pelo algoritmo Perceptron como forma de resolver problemas não linearmente separáveis. Entretanto, somente quando BOSER *et al.* [8] sugeriram a utilização de funções *kernel* em classificadores de ótima margem, o procedimento ficou consagrado e bastante difundido na comunidade de aprendizado de máquinas. Desde então, a partir do desenvolvimento de classificadores *kernel*, abriu-se uma nova perspectiva no uso de classificadores lineares, como o algoritmo Perceptron, em problemas de reconhecimento de padrões.

2.1 Princípio da Minimização do Risco Estrutural

Uma questão fundamental relacionada à teoria do aprendizado de máquinas está associada à capacidade de generalização do classificador. Ou seja, não é suficiente que um classificador seja capaz de separar corretamente somente o conjunto de dados utilizado para o seu aprendizado. É necessário, também, que este seja capaz de discriminar novos dados gerados, evidentemente, a partir da mesma distribuição de probabilidade não conhecida.

Nesse sentido, tornou-se muito importante a descoberta dos classificadores de larga ou máxima margem. O projeto desses classificadores se baseia em um novo princípio denominado Princípio da Minimização do Risco Estrutural (SRM - *Structural Risk Minimization*), [9], que associa a capacidade de um classificador à sua dimensão de Vapnik-Chervonenkis (VC), ao invés do principio de indução tradicional denominado de Princípio da Minimização do Risco Empírico (ERM - *Empirical Risk Minimization*), voltado à minimização de um funcional de risco empírico para uma determinada função de perda.

A eficiência do processo de indução depende da qualidade e tamanho do conjunto de treinamento. Desta forma, uma série de razões pode comprometer o aprendizado indutivo baseado no princípio ERM, como exemplo, a existência de excesso de ruídos, a natureza estocástica do mapeamento e, por fim, a quantidade insuficiente de dados.

O princípio SRM, por outro lado, introduz um novo princípio de indução, aplicado, principalmente, a pequenos conjuntos de treinamento, compromissado com o projeto de classificadores que demonstrem uma melhor capacidade de generalização, buscando um equilíbrio entre a qualidade do erro de treinamento e a capacidade do modelo.

2.2 Teoria da Regularização

A Teoria da Regularização, proposta por TIKHONOV e ARSENIN [10], está relacionada ao estudo da reconstrução de superfícies ou a aproximação de funções, podendo ser aplicada ao problema de generalização. Ela propõe a minimização de um funcional de erro, composto de duas partes: a primeira representada por uma função de perda, que indica o quanto a função se ajusta aos dados, e a segunda, relacionada à natureza da função, que representa a capacidade do modelo. A Teoria da Regularização impõe uma medida de regularidade ou uma imposição de suavidade ao conjunto de funções a serem escolhidas, reduzindo, assim, o espaço de hipóteses relacionadas ao problema.

A idéia central do aprendizado de máquinas é que o aprendizado requer generalização. Portanto, o aprendizado somente será possível sobre a hipótese de que a

superfície a ser reconstruída seja uma superfície suave ou regular, estando compatível com a idéia de compressibilidade e regularidade dos dados. Para um problema de classificação, para uma função de perda definida por $c\left(x_i, y_i, f\left(x_i\right)\right)$ que retrata o risco empírico, pode-se escrever, a minimização do risco ou erro regularizado, como:

$$\text{Min}\left(\sum_i c\left(x_i, y_i, f\left(x_i\right)\right)\right) + \lambda \cdot ||f||K^2 \tag{2.1}$$

O primeiro termo mede a quantidade de erros de classificação associados à função que está sendo aprendida. O segundo termo mede o custo da função f, escolhida como hipótese, se desviar da função ideal segundo um conjunto de restrições conhecidas a priori. Comumente, considera-se a parcela de regularização como o valor da norma da função f representada no espaço de características por uma função *kernel* K semi-definida positiva. O parâmetro λ na equação representa um parâmetro de balanceamento entre as duas medidas.

2.3 Problema de Classificação Binária

Seja um conjunto de dados Z de cardinalidade m, denominado conjunto de treinamento, composto de um conjunto de vetores x_i e de um conjunto de escalares y_i. Cada vetor, rotulado por um valor escalar, está inserido em um espaço de dimensão d, $x_i \in R^d$, chamado de espaço de entrada do problema, representando uma respectiva amostra ou exemplo. Considerando que o valor de cada escalar y_i representa a classe de cada vetor x_i, tem-se para problemas de classificação binária, $y_i \in \{-1, +1\}$ para $i = \{1, \ldots, m\}$. Para problemas linearmente separáveis, um classificador linear será representado no espaço de entrada por um hiperplano, dado pela seguinte equação:

$$f(x) = \langle w, x\rangle + b, \tag{2.2}$$

onde w representa o vetor normal ao hiperplano e b o valor do viés (*bias*).

É possível representar esse classificador em uma forma mais geral tomando-se cada ponto do espaço de entrada em um espaço denominado espaço de características ou $\Phi-$*space*. Neste sentido, pode-se considerar a integração do *bias* da equação em uma componente adicional do vetor w, adicionando, também, uma componente $+1$ no vetor representativo de cada ponto. Assim a equação geral tem a forma:

$$f\left(x\right) = \langle w, \Phi\left(x\right)\rangle \tag{2.3}$$

A resposta do classificador poderá ser obtida através da aplicação de uma função

sinal φ ao valor do discriminante relacionado à equação do hiperplano, ou seja:

$$\varphi\left(f\left(x\right)\right) = +1 \text{ se } f\left(x\right) \geq 0 \text{ ou } \varphi\left(f\left(x\right)\right) = -1 \text{ se } f\left(x\right) < 0 \tag{2.4}$$

2.4 Algoritmo Perceptron

ROSENBLATT [1] propôs um procedimento para a atualização dos pesos de um elemento processador com múltiplas saídas baseando-se na comparação do valor da saída com os valores desejados. O modelo, chamado de Perceptron, é uma máquina para reconhecimento de padrões, aplicado, inicialmente, ao reconhecimento ótico de caracteres. Foi considerado o primeiro algoritmo de aprendizado relacionado a modelos não lineares.

Estruturalmente, o modelo Perceptron é formado por uma camada de entrada, associando cada unidade de *input* a componente de um vetor de dimensão d, e uma camada de saída formada por m unidades. É, portanto, um modelo de Redes Neurais Artificiais com uma única camada de processamento. Os elementos das duas camadas são totalmente interconectados por uma matriz sináptica.

O modelo Perceptron realiza um mapeamento de um espaço de entrada de dimensão d para um espaço de saída de dimensão m reduzida. Na sua forma mais simplificada, o modelo Perceptron é utilizado para problemas de classificação ou reconhecimento de padrões envolvendo somente duas classes e, nesse caso, é suficiente a existência de somente um elemento processador na camada de saída.

Na Figura 2.1 está representada a topologia desse modelo Perceptron.

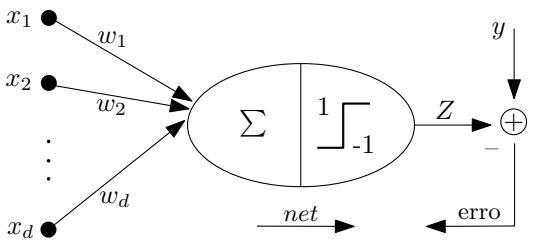


Figura 2.1: Topologia do modelo Perceptron.

O algoritmo desenvolvido por Rosenblatt pode ser utilizado para a determinação do vetor w em um número limitado de iterações. A quantidade de iterações está relacionada à quantidade de atualizações do vetor de pesos e, consequentemente, à quantidade de erros cometidos pelo algoritmo. Neste caso, como o vetor de pesos w, normal ao hiperplano, é determinado com base em sucessivas correções, de modo a minimizar uma função de perda, pode-se dizer que o hiperplano separador é construído de forma iterativa caracterizando um processo de aprendizado definido como *online*.

Para uma determinada amostra do conjunto de treinamento, ocorrerá um erro ou uma classificação incorreta se:

$$y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) < 0 \tag{2.5}$$

Sendo assim, pode-se adotar como função de perda a quantidade de amostras classificadas incorretamente. Essa função, definida como a função de perda $0-1$, é descrita como:

$$J \left(w \right) = \sum_i 1 | \left\{ \varphi \left(f \left(x_i \right) \right) \neq y_i \right\} \tag{2.6}$$

ou

$$J \left(w \right) = \sum_i \text{Max} \left\{ 0, \varphi \left(-y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) \right) \right\}, \left(x_i, y_i \right) \in Z \tag{2.7}$$

Entretanto, sendo essa função constante por partes e, portanto, não diferenciável, torna-se mais apropriado a utilização de uma nova função de perda, linear por partes, dada pela soma negativa de todos os valores funcionais, também chamados de valores de margens, das amostras classificadas incorretamente. Ou seja:

$$J \left(w \right) = \sum_i \text{Max} \left\{ 0, -y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) \right\}, \left(x_i, y_i \right) \in Z, \tag{2.8}$$

tornando possível a utilização do método do gradiente.

Portanto, caso o problema seja linearmente separável no Φ-*space*, para se determinar uma solução que minimize a função de perda em relação ao vetor w, é necessário avaliar o vetor gradiente considerando, somente, a ocorrência das amostras classificadas incorretamente, relacionadas ao conjunto M. Esse processo, aplicado individualmente a cada amostra, resulta na seguinte regra de correção:

$$w_{(t+1)} = w_{(t)} + \eta \cdot \Phi \left(x_i \right) \cdot y_i, \left(x_i, y_i \right) \in Z, \tag{2.9}$$

sendo η a taxa de aprendizado.

Essa versão do método do gradiente que produz, ou computa, um conjunto de valores instantâneos considerando uma única amostra de cada vez, é também chamada de *online* ou estocástica, tendo uma semelhança com o algoritmo LMS (*Least Mean Square*), [11].

2.5 Classificadores *Kernel*

Para obter-se os pontos suporte relacionados à descrição da equação do hiperplano separador é necessário resolver o problema de inequações relacionado ao algoritmo de treinamento do Perceptron na sua forma dual. Para tanto, é preciso representar

o vetor w como uma combinação linear dos pontos do conjunto de treinamento. Tal expansão define um conjunto de escalares positivos, chamados de multiplicadores ou variáveis duais, sendo representados pelo vetor α, $\alpha \in R^m$. Portanto:

$$w = \sum_i \alpha_i y_i \Phi(x_i), \alpha_i \geq 0 \tag{2.10}$$

Substituindo essa representação na equação original, tem-se uma nova forma da função:

$$f(x_i) = \sum_j \alpha_j y_j y_i \langle \Phi(x_i), \Phi(x_j) \rangle \tag{2.11}$$

Sendo assim, para trabalhar-se somente com o conjunto de variáveis duais α, é necessário reescrever a regra de correção do Perceptron na sua forma dual. Para a forma dual do algoritmo, pode-se fazer a mesma atualização, considerando a nova descrição do vetor w:

$$w = \sum_j \alpha_j y_j \Phi(x_j) + \eta \cdot y_i \Phi(x_i), \tag{2.12}$$

resultando, consequentemente, para uma amostra classificada incorretamente, na atualização do respectivo multiplicador com base na expressão:

$$\alpha_i = \alpha_i + \eta \cdot 1 \tag{2.13}$$

Para a atualização do *bias*, representado como uma componente adicional do vetor w, tem-se que considerar o fato de que cada vetor x_i possui um valor adicional $+1$ na sua última componente. Com isso, o valor do *bias* pode ser computado separadamente em um esquema do tipo *online* conforme o vetor de pesos.

Essa representação dual do modelo Perceptron é também chamada de representação dependente dos dados, podendo ser interpretada como um classificador *kernel*. A medida de similaridade entre os dados é computada pelo produto interno dos vetores do espaço de entrada ou dos vetores característicos se for considerada a existência do Φ-*space*.

Por conseguinte, pode-se garantir a separabilidade linear dos dados no espaço apropriado, quando a medida de similaridade dos dados, computada pelos respectivos produtos internos e representada pela matriz *kernel*, é suficiente para permitir a discriminação dos dados em duas classes.

Caso o problema não seja linearmente separável no espaço de entrada, torna-se necessário o mapeamento dos dados para um espaço de características chamado de Φ-*space* e representado por F. Com a realização deste mapeamento, Φ, $\Phi : R^d \times R^d \rightarrow F$, é possível a representação do conjunto de amostras em um espaço de

mais alta dimensão, $x \rightarrow \Phi(x)$, no qual o problema se torna linearmente separável.

Na maior parte dos casos, não é necessário conhecer o tipo de mapeamento ou a função Φ explicitamente. Para tanto, utiliza-se uma função *kernel*, simétrica e semi-definida positiva, definida como $K : R^d \times R^d \rightarrow R$, que atende às condições estabelecidas por MERCER [12]. Os valores dessa função correspondem aos valores do produto interno dos vetores mapeados no espaço de mais alta dimensão, [13].

Neste sentido, torna-se interessante a condução do processo de treinamento no espaço de variáveis duais. Isso porque, a dimensão do espaço de características, a qual define o número de variáveis primais do problema, na maioria dos casos, excede em grande quantidade o número de variáveis duais do mesmo relacionado à cardinalidade do conjunto de treinamento. Por exemplo, para expansões polinomiais, em que os vetores mapeados contêm todas as possibilidades de produtos de determinada ordem, o número de parâmetros primais, que determina a dimensão do espaço de características, é da ordem de d^n, sendo n referente ao grau do polinômio.

Uma topologia de rede pode ser utilizada para a representação dual do Perceptron no espaço de características, ou Φ-*space*, computando-se o produto interno dos vetores característicos. Entretanto, com a introdução de funções *kernel*, pode-se estender essa topologia para uma classe mais geral de classificadores, denominada classificadores *kernel*, mantendo-se os vetores ou amostras do conjunto de treinamento no espaço de entrada. Reescrevendo a equação dual dos classificadores lineares, com o parâmetro *bias*, tem-se a equação discriminante de um classificador *kernel* na forma:

$$f(x_i) = \sum_j \alpha_j y_j y_i K(x_i, x_j) + b, \text{ sendo } K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle \tag{2.14}$$

Como exemplo, mostra-se o desenvolvimento de um classificador que utiliza uma base de funções polinomiais. Considera-se uma expansão quadrática ou mapeamento dos vetores de um conjunto X definido em R^2 para um espaço de características definido em R^3.

$$\begin{aligned} \Phi : R^2(X) &\rightarrow R^3(F) \\ (x_1, x_2) &\mapsto \Phi(x_1, x_2) = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right) \\ K(x, y) &= \langle \Phi(x), \Phi(y) \rangle = \left(x_1^2, \sqrt{2}x_1x_2, x_2^2\right) \cdot \left(y_1^2, \sqrt{2}y_1y_2, y_2^2\right)^T \\ &= \left((x_1, x_2) \cdot (y_1, y_2)^T\right)^2 = (\langle x, y \rangle)^2 \end{aligned} \tag{2.15}$$

2.6 Perceptron de Margem Fixa

2.6.1 Algoritmo Primal

DUDA *et al.* [14] propõem uma versão alternativa para o algoritmo Perceptron, incluindo a utilização de uma regra de incremento variável para uma função de

perda quadrada e um valor fixo de margem ρ, no sentido de adaptar a sua solução ao método de relaxação. Considerando, entretanto, a introdução do parâmetro ρ, a solução do problema consiste na determinação de uma solução viável para o sistema de inequações lineares na forma:

$$y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) \geq \rho \tag{2.16}$$

Entretanto, caso se utilize uma regra de incremento fixo e não seja possível limitar o valor da norma quadrática do vetor w, com a adição da restrição adicional, $||w||_2 = 1$, o sistema de inequações, se linearmente separável, apresentará sempre uma solução viável considerando o crescimento da norma e, consequentemente, do valor do produto interno, para qualquer valor de margem ρ. Para resolver esse problema é necessário estabelecer alguma forma de regularização no sentido de controlar ou de limitar o valor da norma do vetor w.

Assim, LEITE e FONSECA NETO [15] propuseram uma nova formulação para o modelo Perceptron no sentido de garantir que o conjunto de exemplos seja classificado corretamente e, também, guarde uma distância mínima em relação ao hiperplano separador sem limitar diretamente o valor da norma do vetor w. Para tanto, é considerada a restrição de que cada amostra do conjunto de treinamento deve possuir um valor de margem geométrica correspondente superior ou igual ao valor estabelecido como margem fixa, sendo o valor da margem geométrica definido como o valor da margem funcional da respectiva amostra dividido pelo valor da norma euclidiana do vetor w. Nesse sentido, deve-se resolver o seguinte sistema de inequações não lineares para determinado valor de margem fixa representado pelo parâmetro γ_f:

$$y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) / ||w||_2 \geq \gamma_f \tag{2.17}$$

ou

$$y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) \geq \gamma_f \cdot ||w||_2 \tag{2.18}$$

Em função desta modificação, é necessário reescrever a função de erro ou de perda do modelo de forma a tornar possível a obtenção da regra de correção relacionada à aplicação do método do gradiente. A nova função de erro será equivalente à soma dos valores das respectivas margens geométricas dos exemplos que forem menores que o valor da margem fixa, descontado o valor da margem. Ou seja:

$$J \left(w \right) = \sum_i \text{Max} \left\{ 0, \gamma_f - y_i \left(\langle w, \Phi x_i \rangle \right) / ||w||_2 \right\}, (x_i, y_i) \in Z \tag{2.19}$$

$$J \left(w \right) = - \left(\sum_i y_i \left(\langle w, \Phi \left(x_i \right) \rangle \right) - m \cdot \gamma_f \cdot ||w||_2 \right), (x_i, y_i) \in M \text{ e } |M| = m \tag{2.20}$$

Portanto, ao contrário do algoritmo básico de treinamento do Perceptron, considera-se também, como erro, aqueles exemplos que, embora classificados corretamente, não estejam a uma distância mínima, no sentido geométrico, do hiperplano separador. KIVINEN *et al.* [16] definem este tipo de correção como a ocorrência de erros de margem.

A solução do sistema de inequações pode ser considerada como aquela que minimiza a função de erro J em relação aos seus parâmetros primais, representados pelo vetor w que incorpora o valor do *bias*. Com isso, tomando a derivada da função em relação ao vetor w, tem-se a seguinte expressão que define o vetor gradiente:

$$\nabla_w J(w) = (m \cdot \gamma_f \cdot w) / ||w||_2 - \sum_i y_i \Phi(x_i), \tag{2.21}$$

fornecendo, caso ocorra um erro, $y_i(\langle w, \Phi(x_i)\rangle) < \gamma_f \cdot ||w||_2$, a seguinte regra de correção quando aplicado a uma determinada amostra $(x_i, y_i) \in M$:

$$w_{(t+1)} = w_{(t)} - \eta(\gamma_f \cdot w/||w||_2 - \Phi(x_i)\, y_i), \tag{2.22}$$

em que η se refere à taxa de aprendizagem.

Essa equação de correção possui duas interpretações diretas. A primeira, baseada na observação de que o termo $w/||w||_2$ representa o vetor unitário de direção w, sugere que da parcela de correção do vetor normal relacionada ao exemplo apresentado seja retirada uma quantia relativa ao valor da margem fixa. Geometricamente, este novo procedimento de correção pode ser explicado através da Figura 2.2.

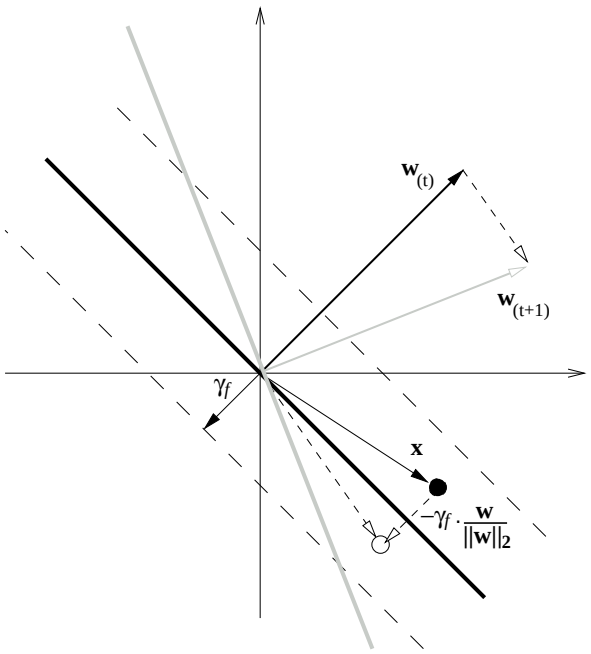


Figura 2.2: Correção do vetor w em função da margem fixa.

A segunda, relacionada à forma alternativa da equação:

$$w_{(t+1)} = w_{(t)} \cdot \left(1 - \left(\eta \cdot \gamma_f\right) / ||w||_2\right) + \eta \cdot \Phi\left(x_i\right) y_i, \tag{2.23}$$

sugere que antes de cada correção do vetor w seja feito um escalonamento redutivo no valor do mesmo proporcional ao valor da margem fixa projetado na sua respectiva direção. É possível afirmar que a forma de regularização empregada, para o controle da norma do vetor, consiste em uma espécie de decaimento no valor dos pesos.

KIVINEN *et al.* [17], em seu trabalho relativo ao algoritmo NORMA e ao aprendizado *online* de métodos *kernel*, definem uma função de perda semelhante, denominada função de perda de margem flexível, compatível com a aplicação do método do gradiente estocástico. É importante lembrar que a função de perda $0-1$, definida como $\text{Max}\left\{0, \varphi\left(-y_i \cdot f\left(x_i\right)\right)\right\}$, é descontínua e não convexa, não sendo, portanto, apropriada a aplicação do método do gradiente. Para um parâmetro de margem ρ, os autores propõem uma nova equação de perda que considera, entretanto, o valor da margem funcional dos exemplos ao invés do valor da margem geométrica:

$$J\left(w\right) = \sum_i \text{Max}\left\{0, \rho - y_i\left(\left\langle w, \Phi x_i \right\rangle\right)\right\}, \left(x_i, y_i\right) \in Z \tag{2.24}$$

Caso ocorra um erro, ou seja, $y_i\left(\left\langle w, \Phi x_i \right\rangle\right) < \rho$, a equação de atualização do vetor de pesos relativa ao algoritmo NORMA é descrita como:

$$w_{(t+1)} = w_{(t)} \cdot \left(1 - \eta \cdot \lambda\right) + \eta \cdot \Phi\left(x_i\right) y_i, \tag{2.25}$$

onde λ se refere ao parâmetro de regularização do funcional associado à equação do risco regularizado, definido por $R_{reg}\left(f\right) = R_{emp}\left(f\right) + \lambda \cdot ||f||^2$, devendo ser escolhido de forma apropriada para cada problema.

A equação de correção é obtida da minimização do risco regularizado tomando-se a derivada em relação à função f, sendo a equação do risco empírico definida pela função de perda de margem flexível. Essa equação é semelhante à equação de correção proposta no algoritmo Perceptron de Margem Geométrica Fixa (*Geometric Margin Fixed Perceptron* – GMFP), no sentido de proporcionar um escalonamento no vetor w como forma de controlar o valor da norma sem a necessidade de limitá-la ao valor unitário. Realizando uma comparação, para o algoritmo GMFP, tem-se $\lambda = \gamma_f / ||w||_2$, devendo o valor da expressão $\eta \cdot \lambda$ pertencer ao intervalo aberto $(0, 1)$.

O Algoritmo 2.1 descreve o pseudocódigo do algoritmo primal relativo ao treinamento do Perceptron de Margem Geométrica Fixa.

Algoritmo 2.1: Perceptron de Margem Geométrica Fixa Primal

Entrada: conjunto de treinamento: $Z = \{(x_i, y_i)\}$ de cardinalidade m;
margem geométrica ou distância de segurança: γ_f;
limite superior no número de épocas: max;
Saída: solução Perceptron: vetor de pesos w^* e bias b;
viabilidade do sistema de inequações: *stop*;

início
  $stop \leftarrow$ falso;
  inicializar $w_{(0)}$;
  $j \leftarrow 0$;
  $b \leftarrow 0$;
  $w \leftarrow \left(w_{(0)}, b\right)$;
  enquanto $j \leq max$ **e** $\neg stop$ **faça**
    $erro \leftarrow$ falso;
    $t \leftarrow 0$;
    para i **de** 1 **até** m **faça**
      se $y_i \left(\langle w, \Phi\left(x_i\right)\rangle\right) < \gamma_f \cdot norma$ **então**
        $w_{(t+1)} \leftarrow w_{(t)} \cdot \left(1 - \left(\eta \cdot \gamma_f\right) / norma\right) + \eta \cdot \Phi\left(x_i\right) y_i$;
        $norma \leftarrow ||w_{(t)}||_2$;
        $t \leftarrow t + 1$;
        $b \leftarrow b + \eta \cdot y_i$;
        $erro \leftarrow$ verdadeiro;
        $w \leftarrow \left(w_{(t+1)}, b\right)$;
      fim se
    fim para
    se $\neg erro$ **então**
      $stop \leftarrow$ verdadeiro;
    fim se
    $j \leftarrow j + 1$;
  fim enqto
fim

2.6.2 Algoritmo Dual

Para que o novo algoritmo tenha o poder de classificação de uma máquina *kernel*, é necessário que o novo modelo Perceptron e o processo de otimização sejam desenvolvidos no plano de variáveis duais. Sendo assim, a equação de correção deve ser modificada, a fim de proporcionar a correção dos respectivos multiplicadores. Considerando a expansão do vetor w em função do conjunto de variáveis duais, tem-se, para a ocorrência de um erro, representado pela condição $y_i \left(\sum_i \alpha_i K\left(x_i, x\right)\right) < \gamma_f \cdot ||w||_2$, o valor do multiplicador associado atualizado pela expressão:

$$\alpha_i = \alpha_i + \eta \cdot 1, \tag{2.26}$$

após a realização de um escalonamento no valor do vetor de multiplicadores, representado por:

$$\alpha_{(t+1)} = \alpha_{(t)} \cdot \left(1 - \left(\eta \cdot \gamma_f\right) / ||w||_2\right), \tag{2.27}$$

para as componentes não nulas do vetor.

A fração $\left(\eta \cdot \gamma_f\right) / ||w||_2$ é responsável pelo decréscimo no valor dos multiplicadores, sendo de fundamental importância na correta determinação dos vetores suporte. Nesse ponto, pode-se anotar um detalhe interessante que ocorre durante o processo de treinamento. O aumento do valor da norma, ao contrário do que se poderia esperar, atua como um fator de decréscimo na taxa de aprendizagem, tornando o processo de correção mais refinado e, portanto, menos instável quando se chega próximo à solução.

A atualização do parâmetro *bias* é feita de modo independente, considerando a existência de uma componente adicional nos pontos de valor igual a $+1$. Essa atualização pode ser *online*, a cada apresentação de uma amostra, ou na forma *batch*, considerando a ocorrência de toda uma época no processo de treinamento.

Se for considerada a execução do algoritmo na sua forma dual, torna-se necessário avaliar o valor da norma do vetor w, $||w||_2$, utilizando somente as variáveis duais. Para tanto, considera-se o cômputo da norma com base na seguinte expressão:

$$w^t w = \left(\sum_i \alpha_i y_i \Phi\left(x_i\right)\right)^t \left(\sum_j \alpha_j y_j \Phi\left(x_j\right)\right) = \sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right) \tag{2.28}$$

$$||w||_2 = \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right)\right)^{1/2} \tag{2.29}$$

Para aumentar a precisão do algoritmo, tornando-o compatível com o processo *online* de aprendizado, e diminuir o esforço de cálculo do valor da norma, é possível adotar uma forma aproximada de atualização do valor da mesma, que considera somente o efeito da modificação, após a ocorrência de cada atualização de um multiplicador α_i, na forma:

$$||w||_2^2 \cong ||w||_2^2 + \sum_j \left(\Delta\alpha_i y_i\right) \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right) \tag{2.30}$$

Durante a atualização do valor da norma, a cada modificação no valor de uma variável dual, deve-se tomar uma atenção especial para que não haja um argumento negativo ou de valor zero relacionado ao valor do produto interno $w^t w$. Entretanto, a garantia de que este valor não seja negativo está associada à propriedade da matriz *kernel* ser semi-definida positiva. Ainda assim, tem-se a possibilidade de o vetor w assumir o vetor nulo. Tal situação pode ser facilmente evitada se for garantida,

igualmente, a propriedade de positividade estrita da matriz *kernel*, ou seja:

$$\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K(x_i, x_j) > 0, \tag{2.31}$$

para quaisquer escalares representados pelos produtos $\alpha_i y_i$ e $\alpha_j y_j$.

É possível assegurar a propriedade de positividade da matriz *kernel* aumentando adequadamente o valor da sua diagonal na forma $K = K + \lambda I$. Essa operação está diretamente relacionada à imposição de uma margem flexível na construção do classificador. Nesse sentido, quanto maior o valor do parâmetro λ, maior poderá ser o valor da margem a ser obtido, observando-se, entretanto, o grau de violação das restrições, possibilitando uma melhor generalização do classificador.

Em uma perspectiva Bayesiana, [18] e [19], onde a matriz *kernel* é vista como a matriz de covariância dos dados, a adição do valor λ às componentes da diagonal principal pode ser interpretada como a soma de uma variância, σ_t^2, relacionada à existência de um ruído no valor de saída, $y_i(\langle w, \Phi(x_i)\rangle)$, de todos os exemplos do conjunto de treinamento. Nessa visão estatística, pode-se considerar que a escolha da função *kernel*, bem como da variância associada, constituem todo o conhecimento a priori que pode ser considerado ou incluído na construção do classificador. Entretanto, como observado por SHAWE-TAYLOR e CRISTIANINI [20], deve haver uma certa precaução na soma de valores às componentes diagonais da matriz *kernel*. A existência de uma diagonal com valores demasiadamente altos em relação aos valores situados fora da diagonal pode causar uma espécie de *overfitting* devido ao fato de o *kernel* representar de forma acentuada o conceito de identidade. Da mesma forma, uma matriz *kernel* com valores muito uniformes, pode levar à ocorrência de *underfitting*.

O Algoritmo 2.2 descreve o pseudocódigo do algoritmo dual relativo ao treinamento do Perceptron de Margem Geométrica Fixa.

2.6.3 Prova de Convergência

A prova de convergência do Perceptron de Margem Geométrica Fixa foi desenvolvida por LEITE e FONSECA NETO [15] tomando por base a prova de convergência original do Perceptron para os casos separáveis, relativo aos trabalhos de BLOCK [2] e NOVIKOFF [3], que estabelecem um limite superior para o número de erros ou correções do algoritmo de treinamento no espaço euclidiano.

Teorema 1 (Teorema de Novikoff sobre a convergência do Perceptron) *Para um conjunto de treinamento* $Z = \{(\Phi(x_1), y_1, \ldots, \Phi(x_m), y_m)\}$ *de cardinalidade* m*, suponha a existência de um vetor solução* w^**, tal que* $\gamma_{w^*} > 0$ *e* $\gamma_{w^*} = Min_i\{(y_i \langle w, \Phi(x_i)\rangle)/||w||_2\}$*, seja a margem geométrica de parada do*

Algoritmo 2.2: Perceptron de Margem Geométrica Fixa Dual

Entrada: conjunto de treinamento: $Z = \{(x_i, y_i)\}$ de cardinalidade m;
margem geométrica ou distância de segurança: γ_f;
limite superior no número de épocas: max;

Saída: solução *kernel* Perceptron: vetor de multiplicadores α^* e bias b;
viabilidade do sistema de inequações: *stop*;

início
  $stop \leftarrow$ falso;
  inicializar α;
  $j \leftarrow 0$;
  $b \leftarrow 0$;
  $norma \leftarrow \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right)\right)^{1/2}$;
  enquanto $j \leq max$ **e** $\neg stop$ **faça**
    $erro \leftarrow$ falso;
    para i **de** 1 **até** m **faça**
      $\alpha_i \leftarrow \alpha_i - \delta$;
    fim para
    para i **de** 1 **até** m **faça**
      se $y_i \left(\sum_j \alpha_j \cdot K\left(x_i, x_j\right)\right) < \gamma_f \cdot norma$ **então**
        $\alpha_i \leftarrow \alpha_i + \eta \cdot 1 + \delta$;
        $norma \leftarrow \left(norma^2 + \sum_j \left(\Delta\alpha_i y_i\right) \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right)\right)^{1/2}$;
        $erro \leftarrow$ verdadeiro;
      fim se
    fim para
    $b \leftarrow b + \Delta\alpha_i y_i$;
    $norma \leftarrow \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right)\right)^{1/2}$;
    se $\neg erro$ **então**
      $stop \leftarrow$ verdadeiro;
    fim se
    $j \leftarrow j + 1$;
  fim enqto
fim

algoritmo relativa à amostra que mais se aproxima do hiperplano separador. Então o número de erros cometidos pelo algoritmo de treinamento do Perceptron será no máximo:

$$t \leq \left(R/\gamma_{w^*}\right)^2, \tag{2.32}$$

onde $R = Max_i ||\Phi\left(x_i\right)||_2$, $i = 1, \ldots, m$.

Prova: deixe w^t *ser o vetor normal do hiperplano separador após a t-ésima atualização. Considerando a regra de correção para um erro cometido na i-ésima*

amostra, pode-se expressar o produto interno $\langle w^*, w^t \rangle$ *como:*

$$\left\langle w^*, w^t \right\rangle = \left\langle w^*, w^{t-1} \right\rangle + \eta \cdot y_i \left\langle w^*, \Phi\left(x_i\right) \right\rangle \tag{2.33}$$

Utilizando a definição da margem geométrica, pode-se escrever:

$$\left\langle w^*, w^t \right\rangle \geq \left\langle w^*, w^{t-1} \right\rangle + \eta \cdot \gamma_{w^*} \cdot ||w||_2 \geq \ldots \geq t \cdot \eta \cdot \gamma_{w^*} \cdot ||w||_2 \tag{2.34}$$

Também, utilizando a regra de correção, tem-se:

$$||w^t||_2^2 = ||w^{t-1}||_2^2 + 2 \cdot \eta \cdot y_i \left\langle w^{t-1}, \Phi\left(x_i\right) \right\rangle + \eta^2 \cdot ||\Phi\left(x_i\right)||_2 \leq \ldots \leq t \cdot \eta^2 \cdot R^2 \tag{2.35}$$

Finalmente, utilizando a inequação de Cauchy-Schwarz, obtém-se:

$$t \cdot \eta \cdot \gamma_{w^*} \cdot ||w||_2 \leq \left\langle w^*, w^t \right\rangle \leq ||w^*||_2 \cdot ||w^t||_2 \leq ||w^*||_2 \cdot t^{1/2} \cdot \eta \cdot R, \tag{2.36}$$

implicando:

$$t \cdot \eta \cdot \gamma_{w^*} \leq \sqrt{t} \cdot \eta \cdot R \text{ } ou \text{ } t \leq \left(R/\gamma_{w^*}\right)^2, \tag{2.37}$$

o que prova o teorema. ■

Teorema 2 (Teorema sobre a convergência do GMFP) *Para um conjunto de treinamento* $Z = \{(\Phi\left(x_1\right), y_1, \ldots, \Phi\left(x_m\right), y_m)\}$ *de cardinalidade* m*, suponha a existência de um vetor solução* w^**, tal que* $\gamma_{w^*} \geq \gamma_f$ *e* $\gamma_{w^*} = Min_i \left\{\left(y_i \left\langle w, \Phi\left(x_i\right)\right\rangle\right) / ||w||_2\right\}$*, seja a margem geométrica de parada do algoritmo relativa à amostra que mais se aproxima do hiperplano separador. Então o número de erros cometidos pelo algoritmo de treinamento do Perceptron de Margem Geométrica Fixa será no máximo:*

$$t \leq \left(R^2 - \gamma_f^2\right) / \left(\gamma_{w^*} - \gamma_f\right)^2, \tag{2.38}$$

onde $R = Max_i ||\Phi\left(x_i\right)||_2$*,* $i = 1, \ldots, m$*.*

Prova: deixe w^t *ser o vetor normal do hiperplano separador após a* t*-ésima atualização. A nova regra de correção do vetor* w *para um erro cometido na* i*-ésima amostra será dada por:*

$$w^t = w^{t-1} - \eta \cdot \left(\gamma_f \cdot w^{t-1}/||w^{t-1}||_2 - \Phi\left(x_i\right) \cdot y_i\right), \tag{2.39}$$

a qual fornece a seguinte equação para expressar o produto interno $\langle w^*, w^t \rangle$*:*

$$\left\langle w^*, w^t \right\rangle = \left\langle w^*, w^{t-1} \right\rangle - \left\langle w^*, w^{t-1} \right\rangle \cdot \eta \cdot \gamma_f / ||w^{t-1}||_2 + \eta \cdot y_i \left\langle w^*, \Phi\left(x_i\right) \right\rangle \tag{2.40}$$

Utilizando a definição da margem geométrica, pode-se escrever:

$$\left\langle w^*, w^t \right\rangle = \left\langle w^*, w^{t-1} \right\rangle - \left\langle w^*, w^{t-1} \right\rangle \cdot \eta \cdot \gamma_f / ||w^{t-1}||_2 + \eta \cdot \gamma_{w^*} \cdot ||w^*||_2, \tag{2.41}$$

ou

$$\left\langle w^*, w^t \right\rangle \geq \ldots \geq t \cdot \eta \cdot \gamma_{w^*} \cdot ||w^*||_2 - \sum_i \left\langle w^*, w^i \right\rangle \cdot \eta \cdot \gamma_f / ||w^i||_2, \tag{2.42}$$

Entretanto, utilizando-se da inequação de Cauchy-Schwarz, tem-se:

$$\sum_i \left\langle w^*, w^i \right\rangle \cdot \eta \cdot \gamma_f / ||w^i||_2 \leq \sum_i ||w^*||_2 \cdot ||w^i||_2 \cdot \eta \cdot \gamma_f / ||w^i||_2 \leq t \cdot ||w^*||_2 \cdot \eta \cdot \gamma_f \tag{2.43}$$

Portanto:

$$\left\langle w^*, w^t \right\rangle \geq t \cdot \eta \cdot ||w^*||_2 \cdot (\gamma_{w^*} - \gamma_f) \tag{2.44}$$

Similarmente, utilizando a nova regra de correção, tem-se:

$$\begin{array}{c} ||w^t||_2^2 = ||w^{t-1}||_2^2 \cdot \left(1 - 2 \cdot \eta\gamma_f / ||w^{t-1}||_2 + \eta^2 \cdot \gamma_f^2 / ||w^{t-1}||_2^2\right) + \\ 2 \cdot \eta \cdot y_i \left\langle w^{t-1}, \Phi\left(x_i\right) \right\rangle > \left(1 - \eta \cdot \gamma_f / ||w^{t-1}||_2\right) + \eta^2 \cdot ||\Phi\left(x_i\right)||_2 \end{array} \tag{2.45}$$

Considerando $R = Max_i ||\Phi\left(x_i\right)||_2$ *e que* $y_i \left\langle w^{t-1}, \Phi\left(x_i\right) \right\rangle < ||w^{t-1}||_2 \cdot \gamma_f$*, obtém-se:*

$$\begin{array}{c} ||w^t||_2^2 \leq ||w^{t-1}||_2^2 - 2 \cdot \eta \cdot \gamma_f / ||w^{t-1}||_2 + \eta^2 \cdot \gamma_f^2 + \\ 2 \cdot \eta \cdot \gamma_f \cdot ||w^{t-1}||_2 - 2 \cdot \eta^2 \cdot \gamma_f^2 + \eta^2 \cdot R^2 \\ ||w^t||_2^2 \leq ||w^{t-1}||_2^2 + \eta^2 \cdot \left(R^2 - \gamma_f^2\right) \end{array} \tag{2.46}$$

Finalmente, utilizando a inequação de Cauchy-Schwarz, obtém-se:

$$t \cdot \eta \cdot ||n^*||_2 \left(\gamma_{w^*} - \gamma_f\right) \leq \left\langle w^*, w^t \right\rangle \leq ||w^*||_2 \cdot ||w^t||_2 \leq ||w^*||_2 \cdot \eta \cdot t \cdot \left(R^2 - \gamma_f^2\right)^{1/2}, \tag{2.47}$$

implicando:

$$t \cdot \left(\gamma_{w^*} - \gamma_f\right) \leq t \cdot \left(R^2 - \gamma_f^2\right)^{1/2} \text{ ou } t \leq \left(R^2 - \gamma_f^2\right) / \left(\gamma_{w^*} - \gamma_f\right)^2, \tag{2.48}$$

o que prova o teorema. ■

É importante na análise do resultado alcançado, verificar que, para uma margem de parada representada por γ_{w^*}, cujo valor máximo seria igual à margem obtida por um Perceptron de margem máxima, o maior valor imposto para a margem fixa deverá ser inferior ao valor da margem máxima, pois do contrário, o número de correções do algoritmo tende para infinito. No outro caso extremo, o qual considera o valor da margem fixa zero, tem-se o mesmo limite de correções da prova original relativa ao algoritmo de treinamento básico do Perceptron.

Também, pode ser observado que, quanto maior o valor da margem fixa, para uma determinada margem de parada, maior será a dificuldade na determinação do hiperplano separador. Problemas de separabilidade difícil, que terminam com baixos valores para a margem de parada terão menor possibilidade de trabalharem com margens fixas de valores elevados. De forma contrária, problemas de separabilidade mais fácil, que terminam com maiores valores para a margem de parada, terão maior possibilidade de solução, permitindo a incorporação de margens fixas de valores superiores.

2.7 Formulação de Máxima Margem

2.7.1 Maximizando o Valor da Menor Distância

Para se obter a máxima margem de um conjunto de treinamento $Z = \{(\Phi(x_i), y_i)\}$, $(x_i, y_i) \in X \times Y$, $X = R^d$ e $Y = \{-1, 1\}$, linearmente separável em um espaço de características F, $\Phi : X \to F$, deve-se maximizar a menor distância existente entre um ponto qualquer deste espaço e o hiperplano separador, segundo uma norma estabelecida. Para uma norma L_2 este problema é equivalente à determinação da maior distância euclidiana, ou distância de separação, entre dois pontos de classes contrárias em relação ao hiperplano separador, [21]. Neste sentido, para toda possível hipótese relacionada à existência de um classificador linear, representada pelos seus respectivos parâmetros, deve-se resolver o seguinte problema de otimização associado à norma euclidiana:

$$\mathrm{Max}_w \mathrm{Min}_{x_i \in X} \{y_i \cdot f(x_i) / ||w||_2\}, \tag{2.49}$$

sendo $f(x_i) = \langle w, \Phi(x_i) \rangle$ a equação que define o classificador linear, considerando a incorporação do parâmetro *bias* ao vetor w.

Considerando os valores da margem funcional e geométrica mínima, para um hiperplano com parâmetro w, respectivamente, como:

$$\gamma = \mathrm{Min}_{x_i \in X} \{y_i \cdot f(x_i)\} \text{ e } \gamma_g = \mathrm{Min}_{x_i \in X} \{y_i \cdot f(x_i) / ||w||_2\} \tag{2.50}$$

Portanto, a máxima margem, no sentido geométrico, relativo à escolha do melhor hiperplano, é definida como $\gamma^* = \mathrm{Max}_w \gamma_g$.

Pelo fato dessa margem ser válida para todos os pontos, tem-se, para qualquer amostra (x_i, y_i) do conjunto de treinamento, a seguinte relação: $y_i \cdot f(x_i) / ||w||_2 \geq \gamma^* \geq \gamma_g$. Sendo assim, pode-se reescrever o problema relativo à maximização da

margem como:

$$\begin{gathered} \mathrm{Max}_w \gamma_g \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) / ||w||_2 \geq \gamma_g \end{gathered} \tag{2.51}$$

2.7.2 Limitando o Valor da Margem Funcional

Para se contornar o problema relacionado à dependência da viabilidade das restrições de classificação em relação à norma do vetor w, VAPNIK e CHERVONENKIS [22], VAPNIK [23] e BOSER *et al.* [8] propõem que o valor da margem funcional relativa à margem geométrica máxima seja fixado em 1, estabelecendo uma relação inversa entre o valor da margem geométrica e a respectiva norma do vetor. Ou seja, para uma norma L_2, tem-se $||w||_2 \cdot \gamma_g = 1$. Sendo assim, é possível resolver o problema de maximização da margem, conhecido como formulação de Máquinas de Vetores Suporte (SVM – *Support Vector Machines*) de margem *hard*, na forma:

$$\begin{gathered} \mathrm{Max}_w \gamma_g \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1 \end{gathered} \tag{2.52}$$

Ou, equivalentemente, dada a existência da relação inversa entre a margem e a norma:

$$\begin{gathered} \mathrm{Min} 1/2 \cdot w \cdot w^T \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1 \end{gathered} \tag{2.53}$$

Essa formulação estabelece na solução ótima, w^*, um valor de margem funcional unitário para todos os pontos que se situarem nos hiperplanos canônicos, ou seja, que estejam mais próximos ao hiperplano separador. Estes pontos são chamados de vetores suporte e são suficientes para descreverem a equação do classificador linear possibilitando um esquema de compressibilidade dos dados e esparsidade da solução, [24]. O valor da margem máxima será dado por $\gamma^* = 1/||w^*||_2$.

No contexto da teoria da reprodução *kernel* em espaços de Hilbert (RKHS), [25], ou das redes de regularização, [26] e [27], é possível a obtenção de um classificador de máxima margem, através da minimização do risco regularizado no espaço de características, ou seja:

$$\mathrm{Min}_f \left(\sum_i c\left(x_i, y_i, f\left(x_i\right)\right) \right) + \lambda \cdot \phi\left(f\left(x\right)\right) \tag{2.54}$$

Considerando $||w||_2 \cdot \gamma_g = 1$, a função de perda individual de um classificador de

máxima margem será dada por:

$$\text{Max}\left\{0, 1 - y_i \cdot f\left(x_i\right)\right\} \tag{2.55}$$

Portanto, para um funcional de regularização relacionado ao valor da norma L_2 das hipóteses candidatas, tem-se:

$$\text{Min}_w \left\{ \sum_i \text{Max}\left\{0, 1 - y_i \cdot f\left(x_i\right)\right\} \right\} + 1/2 \cdot w \cdot w^T \tag{2.56}$$

O que se torna equivalente a:

$$\begin{gathered} \text{Min} 1/2 \cdot w \cdot w^T \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1, \left(x_i, y_i\right) \in Z \end{gathered} \tag{2.57}$$

2.7.3 Limitando o Valor da Norma

Outra possibilidade de solução do problema de maximização da margem consiste na fixação ou limitação do valor da norma do vetor w. Nesse caso, considerando a norma L_2 do vetor unitária, tem-se que resolver o seguinte problema Min-Max:

$$\begin{gathered} \text{Max}_w \text{Min}_{x_i \in X} \left\{y_i \cdot f\left(x_i\right)\right\} \\ \text{Sujeito a} \\ ||w||_2 = 1 \end{gathered} \tag{2.58}$$

Ou, equivalentemente, com a introdução do valor da margem:

$$\begin{gathered} \text{Max}_w \gamma_g \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq \gamma_g \\ ||w||_2 = 1 \end{gathered} \tag{2.59}$$

A particularidade dessa formulação é que, nesse caso, os valores das respectivas margens geométricas e funcionais se equivalem, podendo ser computadas explicitamente pelo algoritmo associado.

2.7.4 Formulação Primal de Mangasarian

MANGASARIAN [28], pioneiramente, propõe uma formulação para o problema de maximização da margem no contexto da separabilidade linear de padrões que é bastante similar à formulação SVM. A formulação, escrita para a solução de um problema de classificação binária, com o respectivo espaço de entrada dividido em

X^+ e X^-, se baseia na solução do seguinte problema de otimização:

$$\begin{gathered} \text{Max } \alpha - \beta \\ \text{Sujeito a} \\ f(x_i) \geq \alpha, \text{ para } x_i \in X^+ \\ f(x_i) \leq \beta, \text{ para } x_i \in X^- \\ ||w||_p = 1 \end{gathered} \tag{2.60}$$

Para uma norma euclidiana, a solução do problema em α, β e w determina uma margem final igual a $(\alpha - \beta)/2$ e equivalente à margem máxima da formulação SVM. A equivalência é facilmente percebida se for fixada a diferença $\alpha - \beta = 2$, correspondente a uma margem funcional de valor unitário, e minimizada a norma euclidiana do vetor w.

Pode-se observar, entretanto, que o objetivo de Mangasarian, ao limitar a norma do vetor, é impedir que o seu valor suba indefinidamente tornando impossível a imposição de uma margem de separação máxima. Vapnik, posteriormente, para amenizar o mesmo problema, estabelece uma relação inversa entre o valor da margem geométrica e a norma, tornando possível a imposição de uma margem máxima sem ter que trabalhar explicitamente com o valor da margem na formulação do problema.

Considerando a restrição não convexa de normalização, $||w||_2 = 1$, de difícil solução, Mangasarian propõe a limitação da norma L_∞ como forma de possibilitar a determinação de uma solução com margem máxima. Definindo a norma L_∞ do vetor w como $||w||_\infty = \text{Max}|w_i|$, para $i = 1, \ldots, d$, e fixando o seu valor igual a 1, o problema relativo à formulação de Mangasarian pode ser resolvido mais facilmente através da solução de $2 \cdot d$ problemas de Programação Linear, onde d se refere à dimensão do espaço direto, ou espaço de entrada, cada um na forma:

$$\begin{gathered} \text{Max } \alpha - \beta \\ \text{Sujeito a} \\ f(x_i) \geq \alpha, \text{ para } x_i \in X^+ \\ f(x_i) \leq \beta, \text{ para } x_i \in X^- \\ -1 \leq |w_i| \leq 1 \\ w_i = 1 \text{ ou } -1, \text{ considerando } i = 1, \ldots, d \end{gathered} \tag{2.61}$$

Assim, para cada possível problema independente, é fixado o valor de uma componente do vetor w, primeiramente igual a 1 e, posteriormente, igual a -1, fazendo com que a norma L_∞ do mesmo permaneça limitada ao valor unitário. A par do trabalho de se resolver $2 \cdot d$ problemas de Programação Linear quando a dimensão do espaço do problema for elevada, a formulação proposta por Mangasarian possui uma característica interessante que está relacionada a não existência da solução

nula, caracterizada pelo fato do vetor w adquirir o valor nulo, [29].

2.7.5 Formulação Dual de Vapnik

BOSER *et al.* [8] propõem para a solução do problema de maximização da margem relacionada à minimização da norma L_2 do vetor w, a maximização da margem no espaço dual com a introdução de um vetor de multiplicadores ou variáveis duais α decorrente da respectiva relaxação das restrições de classificação. Portanto, da formulação SVM, define-se a função lagrangeana na forma:

$$\begin{gathered} L\left(w,b,\alpha\right) = 1/2 \cdot w \cdot w^T - \sum_i \alpha_i \cdot \left(y_i \left(\left\langle w, \Phi\left(x_i\right)\right\rangle + b\right) - 1\right) \\ \text{Sujeito a} \\ \alpha_i \geq 0, i = 1, \ldots, m \end{gathered} \tag{2.62}$$

Computando-se as derivadas parciais em relação ao conjunto de parâmetros primais, w e b, e substituindo, adequadamente, o valor de w na função lagrangeana, tem-se o problema de maximização de margem na forma dual de Wolfe. Ou seja:

$$\begin{gathered} \text{Max } W\left(\alpha\right) = \sum_i \alpha_i - 1/2 \cdot \sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \left\langle \Phi\left(x_i\right), \Phi\left(x_j\right)\right\rangle \\ \text{Sujeito a} \\ \sum_i \alpha_i \cdot y_i = 0 \\ \alpha_i \geq 0, i = 1, \ldots, m \end{gathered} \tag{2.63}$$

A margem geométrica máxima obtida da solução do problema é equivalente a $1/||w||_2$, já que o valor da margem funcional para os vetores suporte será sempre unitário. É importante lembrar que o valor da norma do vetor w pode ser computado diretamente a partir de sua expansão em função do conjunto de vetores suporte. Ou seja, dado que $||w||_2 = \left(w \cdot w^T\right)^{1/2}$, $w = \sum_i \alpha_i \cdot \left(y_i \cdot \Phi\left(x_i\right)\right)$, pode-se computar o valor da norma de w como:

$$||w||_2 = \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \left\langle \Phi\left(x_i\right), \Phi\left(x_j\right)\right\rangle\right)^{1/2} \tag{2.64}$$

2.7.6 Formulação Dual de Shawe-Taylor e Cristianini

SHAWE-TAYLOR e CRISTIANINI [20] propõem uma formulação baseada na fixação da norma L_2, estabelecendo como a margem geométrica o valor da margem funcional dos pontos. Para tanto, torna-se necessária a solução do seguinte

problema de otimização:

$$\begin{gathered} \text{Min}_w - \gamma_g \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq \gamma_g \text{ e } ||w||_2 = 1 \end{gathered} \tag{2.65}$$

Para contornar o problema relativo à não convexidade da restrição de normalização do vetor w, os autores propõem a relaxação do conjunto de restrições, o que torna possível a solução do problema em relação ao conjunto de parâmetros duais, realizando-se, a exemplo da formulação SVM, a maximização da margem no espaço dual. Define-se, portanto, a função lagrangeana na forma:

$$\begin{gathered} L(w, b, \gamma, \alpha, \lambda) = -\gamma - \sum_i \alpha_i \cdot (y_i (\langle w, \Phi(x_i) \rangle + b) - \gamma) + \lambda \cdot \left(w \cdot w^T - 1\right) \\ \text{Sujeito a} \\ \alpha_i \geq 0, i = 1, \ldots, m \end{gathered} \tag{2.66}$$

Computando-se as derivadas parciais em relação ao conjunto de parâmetros primais, w, b e γ, e substituindo, adequadamente, os valores de w e γ na função lagrangeana, observando o fato de que $||w||_2 = 1$, tem-se a função objetivo do problema na forma dual de Wolfe:

$$W(\alpha, \lambda) = -1/4 \cdot \sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle - \lambda \tag{2.67}$$

Considerando que o máximo de $W(\alpha, \lambda)$ em relação ao parâmetro λ fornece:

$$\lambda = 1/2 \cdot \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle \right)^{1/2} \tag{2.68}$$

Deve-se resolver o seguinte problema de otimização quadrática, equivalente ao problema dual da formulação SVM:

$$\begin{gathered} \text{Max } - \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle \right)^{1/2} \\ \text{Sujeito a} \\ \sum_i \alpha_i = 1 \\ \sum_i \alpha_i y_i = 0 \\ \alpha_i \geq 0, i = 1, \ldots, m \end{gathered} \tag{2.69}$$

Pela equivalência da solução dual tem-se, no ponto de ótimo, o valor computado

para a margem geométrica máxima, ou seja:

$$\gamma^* = -L(\alpha^*) = \left(\sum_i \sum_j \alpha_i^* y_i \cdot \alpha_j^* y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle \right)^{1/2}, \tag{2.70}$$

considerando a norma L_2 do vetor w unitária este valor será o mesmo para a margem funcional.

Tomando como base a solução fornecida pela formulação de Shawe-Taylor e Cristianini, pode-se definir o valor da margem geométrica máxima, da formulação SVM, como sendo igual a:

$$\gamma^* = \frac{1}{\left(\sum_i \alpha_i^*\right)^{-1/2}} \tag{2.71}$$

A prova deste resultado, como escrita por SHAWE-TAYLOR e CRISTIANINI [20], é estabelecida com base na equivalência do valor das margens e na escalabilidade das soluções em comparação à formulação SVM.

2.8 Algoritmo de Margem Incremental

2.8.1 Maximização da Margem

Uma nova formulação para o problema de maximização da margem foi proposta por LEITE e FONSECA NETO [15] e desenvolvida a partir de duas constatações importantes. Primeiramente, observando o fato de que, na obtenção da máxima margem, os pontos ou vetores suporte de classes contrárias se encontram à mesma distância do hiperplano separador, ou seja, considerando as margens das classes de rótulos positivo e negativo, tem-se $\gamma^+ = \gamma^-$, onde:

$$\begin{aligned} \gamma^+ &= \text{Min } y_i \cdot f(x_i), \text{ para todo } x_i \in X^+ \\ \gamma^- &= \text{Min } y_i \cdot f(x_i), \text{ para todo } x_i \in X^- \end{aligned} \tag{2.72}$$

Essa condição de igualdade das margens negativa e positiva pode ser facilmente comprovada a partir da interpretação da equação de complementaridade das condições de Karush-Kuhn-Tucker (KKT):

$$\alpha_i \cdot (y_i \cdot f(x_i) - 1) = 0, \tag{2.73}$$

a qual estabelece que os vetores suporte das duas classes, $\alpha_i > 0$, devem possuir o mesmo valor absoluto de margem funcional e, consequentemente, de margem geométrica.

Em segundo lugar, observando a possibilidade da obtenção de soluções de larga margem, em um número finito de correções, na solução do problema do Perceptron

de Margem Geométrica Fixa (GMFP), na forma:

$$y_i \cdot f(x_i) \geq \gamma_f \cdot ||w||_2, \text{ para valores de } \gamma_f < \gamma^* \tag{2.74}$$

Nesse sentido, propõem-se a formulação e solução aproximada do problema de máxima margem, considerando a maximização explícita e direta da margem geométrica. Portanto, deve-se resolver o seguinte problema de otimização:

$$\begin{gathered} \text{Max}_w \gamma_g \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq \gamma_g \cdot ||w||_2 \end{gathered} \tag{2.75}$$

Diferentemente da formulação proposta por Shawe-Taylor e Cristianini, optou-se pela não limitação da norma do vetor w. Nesse caso, o algoritmo computa diretamente o valor da maior margem geométrica, a qual se aproxima suficientemente da margem ótima no sentido de garantir a construção de um classificador de larga margem. Também, não existe uma limitação do valor da margem funcional dos pontos ou vetores suporte que definem a equação do hiperplano separador.

Como pode ser observado na prova de convergência do GMFP, existe uma limitação no crescimento do valor da norma do vetor w impedindo que o mesmo escape para valores muito altos. Assim, os vetores determinados pelas diferentes formulações possuem a mesma direção, derivando o mesmo hiperplano separador, sendo a diferença entre eles tão somente de escalabilidade.

2.8.2 Técnica de Solução

A técnica de solução desenvolvida consiste em uma estratégia de aprendizado incremental, através da qual são obtidas sucessivas soluções do problema do Perceptron de Margem Geométrica Fixa, para valores crescentes de margem. Esse parâmetro inicia com o valor zero, equivalente à solução original do algoritmo Perceptron, e tem seus valores incrementados de forma consistente, até aproximar-se do valor da margem máxima. Ou seja, para um conjunto de valores $\gamma_f \in [0..\gamma^*)$, sendo:

$$\gamma_f^{t+1} > \gamma_f^t, \text{ para } t = 1, \ldots, T-1, \gamma_f^1 = 0, \gamma_f^T \approx \gamma^*, \tag{2.76}$$

soluciona-se, sucessivamente, o problema de inequações não lineares:

$$y_i \cdot f(x_i) \geq \gamma_f \cdot ||w||_2, i = 1, \ldots, m, \tag{2.77}$$

sendo cada solução equivalente à solução do problema do Perceptron de Margem Geométrica Fixa.

Para a atualização, a cada iteração, do valor da margem fixa, adotam-se duas regras, baseadas em uma estratégia de balanceamento, que garantem a convergência para a solução de máxima margem:

Primeira regra: caso a solução do problema forneça as margens, negativa e positiva, diferentes, pode-se dizer que a solução obtida não caracteriza uma solução de máxima margem. Portanto, corrige-se o valor da margem fixa na forma:

$$\gamma_f^{t+1} = \frac{\gamma^+ + \gamma^-}{2}, \tag{2.78}$$

onde γ^+ e γ^- são os valores relacionados, respectivamente, às menores distâncias projetadas dos pontos do conjunto X^+ e X^- ao hiperplano separador da t-ésima iteração.

Pode-se observar que, nesse caso, haverá sempre a garantia de solução do novo problema, já que a nova margem fixa é sempre inferior à margem ótima, ou seja, $\gamma_f^{t+1} = (\gamma^+ + \gamma^-)/2 < \gamma^*$. Tal condição deriva do fato de que se as margens negativa e positiva são desiguais, então a margem total não é máxima, implicando $\gamma^+ + \gamma^- < 2 \cdot \gamma^*$. Também se tem a garantia de convergência, já que a nova margem fixa obtida é superior à margem fixa anterior, ou seja: $\gamma_f^{t+1} > \gamma_f^t$. Tal condição deriva-se da constatação das seguintes relações de exclusividade:

$$\gamma^+ > \gamma_f^t \text{ e } \gamma^- \geq \gamma_f^t \text{ ou } \gamma^+ \geq \gamma_f^t \text{ e } \gamma^- > \gamma_f^t, \tag{2.79}$$

garantindo um incremento no valor da nova margem fixa.

Segunda regra: caso a solução do problema forneça as margens, negativa e positiva, iguais, pode ser que a solução obtida seja uma solução de ótimo local. Portanto, torna-se necessário garantir um acréscimo no valor da nova margem fixa, na forma:

$$\gamma_f^{t+1} = \gamma_f^t + \text{Max}\left\{\Delta, \frac{\gamma^+ + \gamma^-}{2} - \gamma_f^t\right\}, \tag{2.80}$$

sendo Δ uma constante de incremento positiva.

Entretanto, para essa nova forma de atualização, em alguns casos, não se tem mais a garantia de solução do novo problema, já que o novo valor da margem fixa poderá ser igual ou maior que o valor da margem ótima, ou seja, $\gamma_f^{t+1} \geq \gamma^*$. Para a solução desse contratempo, é suficiente a imposição de um número máximo de iterações no número de épocas do algoritmo de treinamento, a partir do qual, caso não haja uma nova solução do problema GMFP, adota-se como margem obtida o valor anterior da margem fixa, relacionado à última solução.

2.8.3 Obtenção da Margem Geométrica

A margem geométrica final estabelecida pelo processo incremental pode ser definida como a margem de parada do algoritmo GMFP na última iteração, $\gamma^* > \gamma_{w^*} \geq \gamma_f^T$, podendo ser computada diretamente tomando-se o valor final do conjunto de multiplicadores, ou seja:

$$\gamma_{w^*} = \left(\sum_i \alpha_i \right)^{-1/2} \approx \gamma^* \tag{2.81}$$

Pode-se observar que, nesse caso, não existe uma relação inversa entre o valor da margem geométrica e o valor da norma quadrática do vetor como na formulação SVM. Assim, o valor da margem funcional dos vetores suporte é equivalente a:

$$y_i \cdot f(x_i) = \gamma_{w^*} \cdot ||w||_2 \approx \gamma^* \cdot ||w^*||_2 \tag{2.82}$$

Também, pode-se computar a margem geométrica final, avaliando-se os pontos suporte e o valor da margem em função do valor final do conjunto de multiplicadores, ou seja, determina-se:

$$||w||_2 = \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle \right)^{1/2} \tag{2.83}$$

e

$$y_i \cdot f(x_i) = y_i \cdot \sum_j \alpha_j y_j \cdot \langle \Phi x_i, \Phi(x_j) \rangle, \tag{2.84}$$

para algum x_i suporte cujo $\alpha_i > 0$, obtendo-se:

$$\gamma_{w^*} = y_i \cdot f(x_i) / ||w||_2 \tag{2.85}$$

2.8.4 Número Total de Correções

O limite no número total de correções pode ser estimado utilizando-se a expressão desenvolvida para o problema GMFP com a introdução do valor da margem fixa relativa à penúltima iteração, $\gamma_f^T < \gamma^*$, sendo equivalente a:

$$t \leq \left(R^2 - \left(\gamma_f^T \right)^2 \right) / \left(\gamma_{w^*} - \gamma_f^T \right)^2 \tag{2.86}$$

Caso o algoritmo seja formulado no sentido de garantir uma α aproximação para o valor da margem, como no algoritmo ALMA_2, para uma norma euclidiana, tem-se o valor da margem fixa escrito na forma:

$$\gamma_f^T = (1 - \alpha) \cdot \gamma_{w^*} \tag{2.87}$$

Substituindo-se na expressão anterior, tem-se o mesmo limite no número de correções obtido por GENTILE [30]:

$$t \leq \left(R^2 - ((1-\alpha)\cdot\gamma_{w^*})^2\right) / \left(\gamma_{w^*} - (1-\alpha)\cdot\gamma_{w^*}\right)^2 \tag{2.88}$$

ou

$$t \leq \left(R^2 - ((1-\alpha)\cdot\gamma_{w^*})^2\right) / \left(\alpha^2\cdot\gamma_{w^*}^2\right) \tag{2.89}$$

Considerando a possibilidade de o algoritmo utilizar, no processo de obtenção de um classificador de larga margem, como solução inicial ou ponto de partida, o conjunto final de multiplicadores obtidos da solução do problema GMFP anterior, pode-se reduzir de forma gradativa a diferença existente entre o valor da margem geométrica final e as sucessivas margens fixas, tornando mais rápida a obtenção, a cada problema, do novo hiperplano separador. Esse procedimento tem uma forte relação com os processos de *boosting*, [31], e de mínimo *overlap*, [32], na medida em que, a cada iteração, um conjunto mais seletivo de padrões informativos, relacionados aos multiplicadores positivos, é corrigido novamente e tem os seus respectivos pesos aumentados.

Pode-se estabelecer, então, um limite pessimista no número total de correções tomando-se o somatório dos limites das correções realizadas por cada problema GMFP. Tal análise se faz necessária já que, a priori, não se sabe o valor da margem ótima responsável pelo estabelecimento do valor da margem fixa da última iteração. Considerando como a margem fixa de um novo problema a margem de parada do problema anterior, tem-se:

$$t \leq \sum_{i=1,\ldots,T} \left(R^2 - \left(\gamma_f^i\right)^2\right) / \left(\gamma_f^{i+1} - \gamma_f^i\right)^2 \tag{2.90}$$

Caso seja considerada uma taxa fixa de correção igual ao valor de Δ, implicando $\gamma_f^{i+1} - \gamma_f^i = \Delta$, tem-se o número final de correções limitado de forma inversa ao valor quadrado de Δ. Ou seja, para valores de margem fixa, $\gamma_f \in (0, \Delta, 2\cdot\Delta, \ldots, \gamma^* - \Delta)$, obtém-se o limite:

$$t \leq \sum_{i=1,\ldots,T} \left(R^2 - \left(\gamma_f^i\right)^2\right) / \Delta^2 \tag{2.91}$$

2.8.5 Pseudocódigo do Algoritmo IMA

O algoritmo de aprendizado (margem) incremental (IMA - *Incremental Margin Algorithm*) possui um *loop* principal relacionado às correções do valor da margem fixa e um procedimento aninhado relativo ao algoritmo de treinamento do Perceptron de Margem Geométrica Fixa, resolvido no seu plano primal ou dual. O Algoritmo 2.3 apresenta a descrição em alto nível do mesmo considerando a chamada, a cada

iteração, do algoritmo de treinamento dual do GMFP no sentido de fornecer uma solução viável até a obtenção do hiperplano separador final. O algoritmo se encerra quando não for mais possível a obtenção de uma solução satisfatória relativa à viabilidade do sistema de inequações não lineares.

Algoritmo 2.3: Algoritmo de Margem Incremental Dual

Entrada: conjunto de treinamento: $Z = \{(x_i, y_i)\}$ de cardinalidade m;
Saída: solução *kernel*: vetor de multiplicadores α^* e bias b;
margem de parada: γ^*;

início
  $\gamma_f \leftarrow 0$;
  enquanto $\neg stop$ **faça**
    DualGMFP$(\alpha, b, \gamma_f, Z, stop)$;
    $norma \leftarrow \left(\sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot K\left(x_i, x_j\right)\right)^{1/2}$;
    $\gamma^+ \leftarrow \mathrm{Min}_{i^+} \left\{ y_i \cdot \sum_j \alpha_j y_j \cdot K\left(x_i, x_j\right) \right\} / norma$;
    $\gamma^- \leftarrow \mathrm{Min}_{i^-} \left\{ y_i \cdot \sum_j \alpha_j y_j \cdot K\left(x_i, x_j\right) \right\} / norma$;
    se $\gamma^+ \neq \gamma^-$ **então**
      $\gamma_f \leftarrow \left(\gamma^+ + \gamma^-\right)/2$;
    senão
      $\gamma_f \leftarrow \left(\gamma^+ + \gamma^-\right)/2 + \Delta$;
    fim se
    $\gamma^* = 1/\left(\sum_i \alpha_i^*\right)^{-1/2}$;
  fim enqto
fim

2.9 Margem Flexível

Quando o problema de classificação binária não for linearmente separável no espaço de características pode-se impor uma variável de relaxação ε_i para cada restrição de classificação. A seguir, será mostrado como esse problema pode ser tratado no contexto da formulação SVM, seguindo o trabalho de CORTES e VAPNIK [33], que derivou a formulação SVM de margem flexível 1–norma e, posteriormente, o trabalho de SMOLA e SCHÖLKOPF [34] que mostrou a formulação SVM de margem flexível 2–norma. Os resultados podem ser estendidos, sem perda de generalização, à formulação GMFP, bem como a variante dual do algoritmo Perceptron.

2.9.1 SVM 1–norma

Nesse caso, a formulação do problema de máxima margem tem a forma:

$$\begin{gathered} \text{Min } -\gamma_g + C \cdot \sum_i \varepsilon_i \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq \gamma_g - \varepsilon_i, \text{ para } i = 1, \ldots, m \\ \varepsilon_i \geq 0 \\ ||w||_2 = 1 \end{gathered} \tag{2.92}$$

O parâmetro C atua como um mediador entre o grau de relaxação ou de violação das restrições, medido pela norma L_1 do vetor de relaxação, e a complexidade do modelo. Caso o parâmetro C tenha um valor bastante elevado, tem-se a formulação SVM original, também chamada de margem fixa. Convém lembrar que BOSER *et al.* [8], no trabalho relacionado ao desenvolvimento de um classificador de máxima margem no espaço dual de variáveis com a utilização de funções *kernel*, não mencionaram a formulação de margem flexível o que somente ocorreu, posteriormente, no trabalho de CORTES e VAPNIK [33], no qual foi introduzida a nomenclatura de vetores suporte.

Anteriormente, a solução de um problema não linearmente separável poderia ser interpretada como a minimização da norma negativa dada à nova relação entre a margem e a norma, $\gamma_g \cdot ||w||_2 = -1$, envolvendo a maximização da norma do vetor w, o que nem sempre conduz a bons resultados. A segunda alternativa existente estava relacionada aos trabalhos de MANGASARIAN [35] e a sua técnica de separação não linear com a utilização de multi-superfícies, que consiste de um processo recursivo que gera uma função discriminante linear por partes.

Considerando a formulação proposta por CORTES e VAPNIK [33], relacionada à minimização da norma quadrática do vetor w e a minimização da norma L_1 do vetor de relaxação, tem-se:

$$\begin{gathered} \text{Min } 1/2 \cdot w \cdot w^T + C \cdot \sum_i \varepsilon_i \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq 1 - \varepsilon_i, \text{ para } i = 1, \ldots, m \\ \varepsilon_i \geq 0 \end{gathered} \tag{2.93}$$

Fornecendo a seguinte função lagrangeana, com a introdução dos vetores de

multiplicadores α e μ:

$$\begin{gathered} L\left(w,b,\varepsilon,\alpha,\mu\right) = 1/2 \cdot w \cdot w^T + C \cdot \sum_i \varepsilon_i + \sum_i \alpha_i \cdot \left(y_i \left(\left\langle w, \Phi\left(x_i\right)\right\rangle + b\right)\right) + \\ \sum_i \alpha_i - \sum_i \alpha_i \cdot \varepsilon_i - \sum_i \mu_i \cdot \varepsilon_i \\ \text{Sujeito a} \\ \alpha_i \geq 0 \text{ e } \varepsilon_i \geq 0, \text{ para } i = 1, \ldots, m \end{gathered} \tag{2.94}$$

Tomando o gradiente da função lagrangeana em relação aos parâmetros primais e substituindo adequadamente os valores dos parâmetros w, b e ε na função lagrangeana, tem-se a formulação SVM de margem flexível 1-norma como a solução do seguinte problema de otimização:

$$\begin{gathered} \text{Max } W\left(\alpha\right) = \sum_i \alpha_i - 1/2 \cdot \sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \left\langle \Phi\left(x_i\right), \Phi\left(x_j\right)\right\rangle \\ \text{Sujeito a} \\ \sum_i \alpha_i y_i = 0 \\ 0 \leq \alpha_i \leq C, \text{ para } i = 1, \ldots, m \end{gathered} \tag{2.95}$$

A interpretação mais direta do parâmetro de controle C é a de que, quanto menor o valor de C, menor será o valor dos multiplicadores associados aos vetores suporte e, portanto, maior será o valor da margem geométrica do classificador, dada a existência de uma relação inversa entre eles, levando a um maior poder de generalização.

A introdução da margem flexível nos algoritmos de treinamento *online*, como o Perceptron dual e o Perceptron de Margem Geométrica Fixa, pode ser feita, a princípio, limitando-se superiormente o valor dos multiplicadores pelo parâmetro de controle C. Entretanto, como é permitido o aumento no valor da norma do vetor normal e, consequentemente, nos valores dos multiplicadores, pode-se constatar que essa forma de flexibilização da margem não é a mais apropriada para esses algoritmos. Como os valores dos multiplicadores são limitados no seu crescimento, a sugestão de GUYON e STORK [32] é que, no processo de treinamento, os exemplos associados às restrições violadas, ou seja, aqueles que possuem $\alpha_i = C$, tenham os seus respectivos valores congelados, devendo ser retirados do processo, tornando possível o encerramento do mesmo com a existência de um hiperplano separador.

2.9.2 SVM 2–norma

Nesse caso, considerando a minimização da norma L_2 do vetor de relaxação, SMOLA e SCHÖLKOPF [34] propuseram a seguinte formulação para o problema de máxima

margem flexível:

$$\begin{gathered} \text{Min } {}^1\!/_2 \cdot w \cdot w^T + C \cdot \sum_i \varepsilon_i^2 \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq 1 - \varepsilon_i, \text{ para } i = 1, \ldots, m \\ \varepsilon_i \geq 0, \text{ para } i = 1, \ldots, m \end{gathered} \tag{2.96}$$

Tomando o gradiente da função lagrangeana em relação aos parâmetros primais e substituindo adequadamente os valores dos parâmetros w, b e ε na função lagrangeana, tem-se a formulação SVM de margem flexível 2–norma como a solução do seguinte problema de otimização:

$$\begin{gathered} \text{Max } W(\alpha) = \sum_i \alpha_i - {}^1\!/_2 \cdot \sum_i \sum_j \alpha_i y_i \cdot \alpha_j y_j \cdot \langle \Phi(x_i), \Phi(x_j) \rangle - \sum_i \alpha_i^2 / {}_{2C} \\ \text{Sujeito a} \\ \sum_i \alpha_i y_i = 0 \\ \alpha_i \geq 0, \text{ para } i = 1, \ldots, m \end{gathered} \tag{2.97}$$

Fazendo $\lambda = {}^1\!/_C$, pode-se redefinir o problema de margem flexível segundo a norma L_2 como a soma de uma constante ou parâmetro de regularização a diagonal da matriz *kernel* cujo valor é inversamente proporcional ao valor do parâmetro C. De fato, esse procedimento, que consiste em acrescentar um valor positivo à diagonal da matriz *kernel* do ponto de vista algorítmico, tem como efeito tornar a matriz em questão definida positiva, tornando o problema de classificação mais bem condicionado.

A utilização dessa abordagem para a implementação de um classificador de margem flexível pode ser feita diretamente nos algoritmos *online* de treinamento que utilizam a matriz *kernel*. Na solução do problema relacionado ao GMFP, esse procedimento, além garantir a solução de problemas não linearmente separáveis, tem uma utilidade adicional, pois, sendo a matriz *kernel* definida positiva, pode-se garantir a não existência de um vetor w com a norma nula. É importante ressaltar que essa forma de flexibilização da margem, por alterar os valores da diagonal da matriz *kernel*, modifica a característica linear do classificador, não garantindo mais a existência de um hiperplano separador no espaço de entrada ou de características.

2.10 Formulações Arbitrárias

2.10.1 Formulação L_p

Uma importante flexibilidade, a exemplo dos algoritmos NORMA e ALMA$_p$, fornecida pelo algoritmo IMA está no fato de poder-se trabalhar livremente com qualquer norma diferenciável relacionada ao vetor w. O algoritmo foi denominado de IMA$_p$, [36]. Para diferentes valores de margens fixas, considera-se a solução de sucessivos

problemas na forma:

$$y_i \cdot f\left(\Phi\left(x_i\right)\right) \geq \gamma_f \cdot ||w||_q, \tag{2.98}$$

no sentido de minimizar a norma L_q, dada por $||w||_q = \left(\sum_i |w_i|^q\right)^{1/q}$, do vetor w e de se estabelecer uma solução de margem L_p, baseando-se no fato de que, segundo MANGASARIAN [37], as normas conjugadas p e q satisfazem a relação $1/p + 1/q = 1$.

Nesse caso, a função de perda do modelo toma a forma geral:

$$J\left(w\right) = -\left(\sum_i y_i\left(\langle w, \Phi\left(x_i\right)\rangle\right) - m \cdot \gamma_f \cdot ||w||_q\right) \tag{2.99}$$

Tomando, pois, a derivada da função em relação ao vetor w, tem-se a seguinte expressão que define o vetor gradiente:

$$\nabla_w J\left(w\right) = \left(\left(m \cdot \gamma_f \cdot \varphi\left(w_i\right) \cdot |w_i|^{q-1}\right) / ||w||_q^{q-1}\right) - \sum_i y_i \Phi\left(x_i\right), \tag{2.100}$$

fornecendo, caso ocorra um erro, $y_i\left(\langle w, \Phi\left(x_i\right)\rangle\right) < \gamma_f \cdot ||w||_q$, a seguinte regra de correção quando aplicado a uma determinada amostra $(x_i, y_i) \in M$:

$$w_{(t+1)} = w_{(t)} - \eta\left(\left(\gamma_f \cdot \varphi\left(w_i\right) \cdot |w_i|^{q-1}\right) / ||w||_q^{q-1} - y_i \cdot \Phi\left(x_i\right)\right) \tag{2.101}$$

onde η se refere à taxa de aprendizagem.

2.10.2 Formulação L_∞

Seja a minimização da norma L_1 do vetor w, definindo um hiperplano separador com margem L_∞. Ou seja, a distância computada dos pontos ao hiperplano separador é tal que maximiza o valor da maior componente do vetor normal. Essa variante é aconselhável se for necessária a obtenção de soluções mais esparsas em relação às componentes do vetor w, como no processo de seleção de características. Nesse caso, é possível constatar que a solução, ou hiperplano, proveniente da formulação L_∞ se posiciona sempre quase perpendicular ao eixo da maior componente, tornando-se dependente dessa coordenada. Tal afirmativa, [38], pode ser evidenciada pelo fato de que a projeção L_∞, considerando somente a maior coordenada, pode desprezar as coordenadas com valor nulo sem afetar o valor da distância projetada.

Ainda segundo ROSSET *et al.* [38], é possível estabelecer o relacionamento entre os valores de margens de duas normas distintas pelas respectivas definições, ou seja, para o caso de escolhermos L_1 e L_2:

$$y_i \cdot f\left(x_i\right) / ||w||_1 = y_i \cdot f\left(x_i\right) / ||w||_2 \cdot \left(||w||_2 / ||w||_1\right) \tag{2.102}$$

Pode-se, assim, observar que a margem L_∞, relacionada à margem conjugada, tende a ser maior quando a relação entre as normas for maior que a unidade, o que ocorre, geralmente, quando o vetor w for mais esparso, o que justifica o seu emprego no problema de seleção de características. Para a definição de um hiperplano separador com margem L_∞, seria suficiente a solução do sistema de inequações:

$$y_i \cdot f\left(\Phi\left(x_i\right)\right) \geq \gamma_f \cdot ||w||_1, i = 1, \ldots, m, \tag{2.103}$$

para valores crescentes de γ_f.

Entretanto, devido à descontinuidade da função relacionada à norma L_1 e o fato de que esse problema envolve a minimização explícita da norma L_1 do vetor w, poderia se optar pela solução do problema de otimização:

$$\mathrm{Max}_w \mathrm{Min}_{x_i \in X} \left\{y_i \cdot f\left(x_i\right) / ||w||_1\right\} \tag{2.104}$$

Todavia, considerando o estabelecimento de uma margem funcional de valor unitário, ou seja:

$$\mathrm{Min}_{x_i \in X} \left\{y_i \cdot f\left(x_i\right)\right\} = 1, \tag{2.105}$$

obtém-se o seguinte problema equivalente:

$$\begin{array}{c} \mathrm{Min}\ ||w||_1 \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1 \end{array} \tag{2.106}$$

Substituindo o valor da norma L_1 do vetor w, tem-se:

$$\begin{array}{c} \mathrm{Min}\ \sum_i |w_i| \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1 \end{array} \tag{2.107}$$

Observando o fato de que $w = w^+ - w^-$ implica em $|w_i| = w_i^+ + w_i^-$, KECMAN e HADZIC [39] propõem a solução do problema acima na forma:

$$\begin{array}{c} \mathrm{Min}\ \sum_i \left(w_i^+ + w_i^-\right) \\ \text{Sujeito a} \\ y_i \cdot f\left(x_i\right) \geq 1, w_i^+ \geq 0 \text{ e } w_i^- \geq 0, \end{array} \tag{2.108}$$

sendo $f\left(x_i\right) = \langle w^+ - w^-, \Phi\left(x_i\right)\rangle$.

2.10.3 Formulação L_1

De outra forma, para a minimização da norma L_∞ do vetor w, definindo um hiperplano separador com margem L_1, ou seja, a distância dos pontos ao hiperplano separador é tal que maximiza o somatório em módulo das componentes do vetor normal, seria suficiente a solução do sistema de inequações:

$$y_i \cdot f(x_i) \geq \gamma_f \cdot ||w||_\infty, i = 1, \ldots, m, \tag{2.109}$$

para valores crescentes de γ_f.

Como não é possível a obtenção da derivada da norma L_∞, pode-se, inicialmente, considerar a solução proposta por Mangasarian na seção 2.7.4, e que limita a norma L_∞ do vetor w através da introdução da restrição $||w|| = 1$. Entretanto, ao invés de se ter que solucionar vários problemas de Programação Linear, PEDROSO e MURATA [40] propõem uma formulação alternativa para a definição de um hiperplano separador com margem L_1, que culmina na solução de um único problema de Programação Linear. Baseado no fato de que: $||w|| = \text{Max}_i|w_i|$, o problema de maximização da margem pode ser reformulado como:

$$\text{Max}_w \text{Min}_{x_i \in X} \{y_i \cdot f(x_i) / \text{Max}_i|w_i|\} \tag{2.110}$$

Limitando o valor da margem funcional, de modo que $\text{Min}_{x_i \in X} \{y_i \cdot f(x_i)\} = 1$, o problema de otimização se torna equivalente a:

$$\begin{gathered} \text{Max}_w \{1/\text{Max}_i|w_i|\} \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq 1 \end{gathered} \tag{2.111}$$

ou

$$\begin{gathered} \text{Min } z \\ \text{Sujeito a} \\ y_i \cdot f(x_i) \geq 1, z \geq +w_i \text{ e } z \geq -w_i, \end{gathered} \tag{2.112}$$

Mesmo com a possibilidade de se resolver um único problema de Programação Linear, percebe-se que a condução do processo de otimização no espaço primal para as formulações L_1 e L_∞, torna inviável a solução de problemas de altíssima dimensão, já que a matriz de coeficientes possuirá posto relacionado à quantidade destes parâmetros. Também, a não utilização de funções *kernel*, decorrente da não condução do processo de otimização no espaço de variáveis duais, torna proibitiva a solução de problemas não linearmente separáveis que necessitam do uso de superfícies de decisão não lineares e do estabelecimento de um espaço de características. Nesse sentido, verifica-se como oportuna a utilização da formulação livre

de limites, tomando-se valores adequados, na solução do problema de maximização da margem para uma norma arbitrária.

2.11 Otimização Sequencial Mínima

Nesta seção será apresentada uma descrição do algoritmo de PLATT [41], conhecido como SMO (*Sequential Minimal Optimization*), para a solução do problema SVM segundo a formulação dual de Vapnik.

2.11.1 Escolha das Variáveis

O procedimento SMO envolve um *loop* iterativo onde, a cada iteração, é escolhido um par de variáveis duais, α_1 e α_2, para a solução analítica de um subproblema de otimização quadrática na sua forma mínima, em R^2.

Como visto por OSUNA *et al.* [42], a solução ótima de um subproblema pode ser alterada resultando em uma melhora da função objetivo caso a variável que ingresse no conjunto ativo esteja violando uma condição de solução ótima de um subproblema pode ser alterada resultando em uma melhora da função objetivo caso a variável que ingresse no conjunto ativo esteja violando uma condição de Karush-Kuhn-Tucker. Nesse caso, deve-se sempre, na escolha das duas variáveis, optar primeiramente por uma que esteja violando as condições de otimalidade. Nesse caso, deve-se sempre, na escolha das duas variáveis, optar primeiramente por uma que esteja violando as condições de otimalidade.

Sendo o erro $E_i = w \cdot x_i + b - y_i$, pode-se reescrever a condição KKT relacionada a complementaridade na forma:

- Se $\alpha_i = 0$, então $y_i \cdot E_i \geq 0$, significando que o ponto está fora das margens;
- Se $0 < \alpha_i < C$, então $y_i \cdot E_i \approx 0$, significando que o ponto está sobre uma margem;
- Se $\alpha_i = C$, então $y_i \cdot E_i \leq 0$, significando que o ponto está dentro da margem.

Para isso, deve-se escolher como primeira variável aquele multiplicador α_i que está presente em uma margem, $0 < \alpha_i < C$, e que não satisfaça a condição associada, portanto $y_i \cdot E_i > \epsilon$, para uma determinada constante de erro. Geralmente escolhe-se $\epsilon = 0,001$.

Platt sugere que a escolha das variáveis seja feita no sentido de maximizar a expressão: $|E_2 - E_1|$, associando o erro E_2 à primeira escolha (variável α_2), maximizando desta forma o passo relacionado à solução do subproblema. Caso E_2 seja

positivo, o algoritmo escolhe como segunda variável, variável α_1, um exemplo associado ao menor erro E_1. Caso o E_2 seja negativo, o algoritmo escolhe para segunda variável, um exemplo associado ao maior erro E_1.

Na escolha da segunda variável é implementada a seguinte hierarquia de escolhas:

- Inicialmente o algoritmo testa para todos os exemplos na margem ou não limitados o exemplo que maximize a expressão: $|E_2 - E_1|$;
- Caso não ocorra melhora da função, o algoritmo testa sequencialmente, a partir de um ponto aleatório, todo o conjunto de exemplos na margem ou não limitados, até que um deles proporcione uma melhora na função;
- Caso ainda não tenha sido obtida essa melhora, o algoritmo finalmente testa sequencialmente, a partir de um ponto aleatório, todo o conjunto de exemplos limitados, até que um deles proporcione uma melhora na função;
- Se essa melhora for obtida, o conjunto retorna com sucesso a escolha da primeira variável considerando somente as variáveis não limitadas, a fim de obter um novo subproblema. Caso contrário o algoritmo retorna com fracasso e tenta, novamente, a escolha da primeira variável considerando todo o conjunto de treinamento;

O algoritmo mantém um vetor de *cache* relacionado aos valores dos erros E_i. Desta forma, antes de ser computado um erro, é verificado se o mesmo já está armazenado na memória. A atualização do vetor de *cache* é feito ao final de cada iteração.

2.11.2 Solução do Subproblema de Otimização

É possível a solução analítica do subproblema de otimização envolvendo somente duas variáveis. Para tanto, primeiramente, é necessário reescrever a função objetivo na forma dual de Wolfe e as respectivas restrições em função das variáveis escolhidas α_1 e α_2:

$$\begin{gathered} L(\alpha) = \alpha_1 + \alpha_2 - s \cdot K(x_1, x_2)\,\alpha_1\alpha_2 - {}^1\!/_2 \cdot K(x_1, x_1)\,\alpha_1^2 - \\ {}^1\!/_2 \cdot K(x_2, x_2)\,\alpha_2^2 - y_1\alpha_1 v_1 - y_2\alpha_2 v_2 - cte \\ s = y_1 y_2 \\ v_i = \textstyle\sum_{j \neq 1,2} y_j \alpha_j \cdot K(x_i, x_j) \\ \text{Sujeito a} \\ y_1\alpha_1 + y_2\alpha_2 = -\textstyle\sum_{i \neq 1,2} \alpha_i y_i \\ 0 \leq \alpha_1 \leq C \\ 0 \leq \alpha_2 \leq C \end{gathered} \tag{2.113}$$

É interessante observar que esse problema possui uma solução que pode ser facilmente construída em R^2, observando que, as restrições de canalização ou limite delimitam uma região viável a um quadrado enquanto que a restrição de igualdade define a equação de uma reta, portanto a escolha dos multiplicadores pertencerá a algum ponto da reta, visto como uma linha diagonal, observando a região delimitada. A utilização de dois multiplicadores é necessária visto que a utilização de somente um multiplicador não permitiria a satisfação da restrição de igualdade a cada passo.

O método de solução consiste em computar o valor da primeira variável considerando o gradiente da função objetivo e determinando um máximo ao longo da direção da reta relativa a restrição de igualdade observando a região viável delimitada pelo quadrado.

Em seguida, a segunda variável é determinada em função do valor da primeira. Para tanto, inicialmente, estabelece-se uma relação entre estas variáveis. Definindo:

$$-\sum_{i\neq 1,2} \alpha_i y_i = \gamma, \tag{2.114}$$

tem-se:

$$\begin{array}{rcl} y_1 = y_2 & \rightarrow & \alpha_1 + \alpha_2 = \gamma \\ y_1 \neq y_2 & \rightarrow & \alpha_1 - \alpha_2 = \gamma, \end{array} \tag{2.115}$$

resultando em:

$$\alpha_1^t = \gamma - s \cdot \alpha_2^t \tag{2.116}$$

Considerando o fato de que, antes da atualização, $\gamma = y_1\alpha_1^t + y_2\alpha_2^t = \alpha_1^t + s \cdot \alpha_2^t$, tem-se:

$$\alpha_1^t = \alpha_1^{t-1} + s\left(\alpha_2^{t-1} - \alpha_2^t\right) \tag{2.117}$$

A obtenção da primeira variável (α_2) é feita computando-se o gradiente da função objetivo.

Inicialmente, deve-se expressar a função objetivo somente em relação a variável α_2, considerando $\alpha_1 = \gamma - s \cdot \alpha_2$. Tomando a função *kernel* $K(x_i, x_j) = K_{ij}$, e realizando a substituição de α_1, pode-se reescrever a função na forma:

$$\begin{array}{c} L(\alpha) = \gamma - s \cdot \alpha_2 + \alpha_2 - {}^1\!/_2 \cdot K_{11}(\gamma - s \cdot \alpha_2)^2 - {}^1\!/_2 \cdot K_{22}\alpha_2^2 - \\ s \cdot K_{12}(\gamma - s \cdot \alpha_2)\alpha_2 - y_1(\gamma - s \cdot \alpha_2)v_1 - y_2\alpha_2 v_2 + cte \end{array} \tag{2.118}$$

Considerando $\eta = 2K_{12} - K_{11} - K_{22}$ e introduzindo os valores dos erros $E_1 = wx_1 + b - y_1$ e $E_2 = wx_2 + b - y_2$, pode-se, após uma sequência de operações algébricas, chegar à expressão:

$$L = {}^1\!/_2 \cdot \eta\alpha_2^2 + \left(y_2\left(E_1^{t-1} - E_2^{t-1}\right) - \eta\alpha_2^{t-1}\right)\alpha_2 + cte \tag{2.119}$$

Computando as derivadas de primeira e segunda ordem da função em relação a α_2, tem-se:

$$\frac{dL}{d\alpha_2} = \eta\alpha_2 + y_2\left(E_1^{t-1} - E_2^{t-1}\right) - \eta\alpha_2^{t-1} \text{ e } \frac{d^2L}{d^2\alpha_2} = \eta \tag{2.120}$$

Fazendo ${}^{dL}/_{d\alpha_2} = 0$, tem-se:

$$\alpha_2^t = \alpha_2^{t-1} + \frac{y_2\left(E_1^{t-1} - E_2^{t-1}\right)}{\eta} \tag{2.121}$$

Para que α_2 seja um ponto estacionário é necessário que $\eta \leq 0$. De fato esta condição é satisfeita, visto que:

$$\eta = 2K_{12} - K_{11} - K_{22} = -\left(x_2 - x_1\right)^T \cdot (x_2 - x_1) = -||x_2 - x_1||_2 \leq 0 \tag{2.122}$$

Caso η seja menor do que zero, a equação de atualização de α_2 fornece um ponto de máximo irrestrito ao longo da reta $\gamma = \alpha_1 + s \cdot \alpha_2$. Portanto, para a viabilidade do novo ponto ser mantida, deve-se observar as restrições de limite $0 \leq \alpha_2 \leq C$, ou seja:

Se $s = 1$, então $\alpha_1 + \alpha_2 = \gamma$, sendo $\gamma = \alpha_1^t + s \cdot \alpha_2^t$, tem-se:

- Caso $\gamma \geq C$, então $\text{Max}\left(\alpha_2\right) = C$ e $\text{Min}\left(\alpha_2\right) = \gamma - C$;
- Caso $\gamma < C$, então $\text{Max}\left(\alpha_2\right) = \gamma$ e $\text{Min}\left(\alpha_2\right) = 0$.

Se $s = -1$, então $\alpha_1 - \alpha_2 = \gamma$, tem-se:

- Caso $\gamma \geq 0$, então $\text{Min}\left(\alpha_2\right) = 0$ e $\text{Max}\left(\alpha_2\right) = -\gamma$;
- Caso $\gamma < 0$, então $\text{Min}\left(\alpha_2\right) = -\gamma$ e $\text{Max}\left(\alpha_2\right) = C$.

É comum essas equações serem estabelecidas em função de dois limites, sendo o limite inferior L e o limite superior H. Neste caso, para $s = 1$, faz-se:

$$L = \text{Max } \{0, \gamma - C\} \text{ e } H = \text{Min } \{C, \gamma\} \tag{2.123}$$

De outra forma, para $s = -1$, faz-se:

$$L = \text{Max } \{0, -\gamma\} \text{ e } H = \text{Min } \{C, C + \alpha_2 - \alpha_1\}, \tag{2.124}$$

obtendo, portanto, as novas equações de fixação:

$$\alpha_2 = L \text{ se } \alpha_2 < L \text{ e } \alpha_2 = H \text{ se } \alpha_2 > H \tag{2.125}$$

Caso $\eta = 0$, é necessário avaliar a função objetivo, ou seja, o valor da reta no final dos dois pontos referentes aos valores de $\alpha_2 = L$ ou $\alpha_2 = H$, tomando para α_2

o valor que fornece o máximo da função. Em outras palavras, computa-se:

$$L_{obj}(\alpha_2) = {}^1/_2 \cdot \eta\alpha_2^2 + \left(y_2\left(E_1^{t-1} - E_2^{t-1}\right) - \eta\alpha_2^{t-1}\right)\alpha_2 + cte, \tag{2.126}$$

para $\alpha_2 = L$ e $\alpha_2 = H$, e, com isso, é definido:

$$\alpha_2^t = \text{Arg Max}\{L(\alpha)\} \tag{2.127}$$

2.11.3 Atualização de Parâmetros e *Cache*

Ao final de cada iteração, deve-se atualizar ou computar o vetor w, o vetor de erro E, considerando os erros dos multiplicadores α_i não limitados e que não se envolveram no processo de otimização, e o *bias* b da equação do hiperplano. Convém observar que todo multiplicador não limitado envolvido no processo de otimização tem seu valor de erro igualado a zero.

O armazenamento do vetor de erros na memória, reduz o esforço computacional na solução do subproblema de otimização. Logo, sempre que for necessário a utilização de seus valores, verifica-se primeiramente se o mesmo já não se encontra calculado. Caso não esteja, é feita a sua avaliação *online*.

Primeiramente, para o vetor w, tem-se:

$$w = w^{t-1} + y_1\left(\alpha_1 - \alpha_1^{t-1}\right)\cdot x_1 + y_2\left(\alpha_2 - \alpha_2^{t-1}\right)\cdot x_2 \tag{2.128}$$

Em seguida, para o vetor de erro E, tem-se:

$$E_i = E_i^{t-1} + y_1\left(\alpha_1 - \alpha_1^{t-1}\right)\cdot K(x_1, x_i) + y_2\left(\alpha_2 - \alpha_2^{t-1}\right)\cdot K(x_2, x_i) + b^{t-1} - b \tag{2.129}$$

Finalmente, para o *bias* b, tem-se:

$$\begin{aligned} b_1 &= E_1 + y_1\left(\alpha_1 - \alpha_1^{t-1}\right)\cdot K(x_1, x_1) + y_2\left(\alpha_2 - \alpha_2^{t-1}\right)\cdot K(x_1, x_2) + b^{t-1} \\ b_2 &= E_2 + y_1\left(\alpha_1 - \alpha_1^{t-1}\right)\cdot K(x_1, x_2) + y_2\left(\alpha_2 - \alpha_2^{t-1}\right)\cdot K(x_2, x_2) + b^{t-1} \\ b &= {}^{b_1 + b_2}/_2 \end{aligned} \tag{2.130}$$

Os valores da função *kernel* são computados na fase inicial do algoritmo e armazenados em uma matriz K, de dimensão $m \times m$, sendo m o número de exemplos do conjunto de treinamento. Caso o valor de m seja muito elevado, é necessário a avaliação *online* dos valores da função *kernel*, demandando um maior custo computacional.

2.11.4 Pseudocódigo do Algoritmo SMO

O Algoritmo 2.4 apresenta uma descrição em alto nível do algoritmo SMO seguindo a metodologia de cálculo vista anteriormente. Foram considerados dois módulos principais, sendo o primeiro módulo relacionado ao *loop* principal que controla a chamada do módulo auxiliar e o processo de convergência e o segundo relacionado à escolha das variáveis α, juntamente com a solução do subproblema de otimização e atualização dos parâmetros e *cache*.

Algoritmo 2.4: Otimização Sequencial Mínima

Entrada: conjunto de treinamento: $Z = \{(x_i, y_i)\}$ de cardinalidade m;
Saída: solução *kernel*: vetor de multiplicadores α^* e bias b;
margem de parada: γ^*;

```
1  início
2      mudanca ← 0;
3      loop ← 1;
4      para i de 1 até m faça
5          α_i ← 0;
6      fim para
7      b ← 0;
8      enquanto mudanca > 0 ou loop = 1 faça
9          mudanca ← 0;
10         se loop então
11             para i de 1 até m faça
12                 mudanca ← mudanca + ExaminaAmostra(i);
13             fim para
14         senão
15             para i ∈ suporte faça
16                 mudanca ← mudanca + ExaminaAmostra(i);
17             fim para
18         fim se
19         se loop = 1 então
20             loop ← 0;
21         senão se mudanca = 0 então
22             loop ← 1;
23         fim se
24     fim enqto
25 fim
```

Função ExaminaAmostra(i)

```
1  início
2      y2 ← target(x2);
3      α2 ← multiplicador de Lagrange para i; // α2 que viola KKT
4       E2 ← w · x2 + b − y2;
5      r2 ← E2 · y2;
6      se (r2 < −tolerancia e α2 < C) ou (r2 > tolerancia e α2 > 0) então
7          se existe 0 < αi < C então
8              se E2 > 0 então
9                  α1 = Arg Min_{i≠α2} E1 (w · xi − yi);
10             senão
11                 α1 = Arg Max_{i≠α2} E1 (w · xi − yi);
12             fim se
13             se houve melhora na função objetivo com o conjunto α1, α2 então
14                 retorna 1;
15             fim se
16         fim se
17         para todo i fora da margem, iniciando em um ponto aleatório faça
18             se houve melhora na função objetivo com o conjunto α1, α2 então
19                 retorna 1;
20             fim se
21         fim para todo
22         para todo i na margem, iniciando em um ponto aleatório faça
23             se houve melhora na função objetivo com o conjunto α1, α2 então
24                 retorna 1;
25             fim se
26         fim para todo
27     fim se
28     retorna 0;
29 fim
```

Capítulo 3

Seleção de Características

If variable elimination has not been sorted out after two decades of work assisted by high-speed computing, then perhaps the time has come to move on to other problems.

R. L. Plackett

O processo de seleção de características (*feature selection* ou *feature subset selection*), ou seleção de atributos, tem sido um tradicional tópico de pesquisa desde os anos 60, [43], e 70, [44]. O processo baseia-se na seleção, segundo algum critério, de um subconjunto do conjunto original de características do problema que produza os mesmos (ou quase mesmos) resultados. A técnica é aplicada nas mais diversas áreas como, por exemplo, reconhecimento de padrões, mineração de dados, aprendizado de máquina e programação matemática.

Entre as vantagens em se utilizar seleção de características, é possível citar o fato de que, após a sua aplicação, a dimensionalidade do espaço representativo do problema é reduzida. Assim, o processo remove atributos redundantes (altamente correlacionados e que não agregam informação) e irrelevantes (que não contêm informações úteis). Sua aplicação traz resultados imediatos. Dentre eles, pode-se destacar:

- Melhoramento da qualidade dos dados;
- Extração de conhecimento mais compreensível;
- Obtenção de uma maior velocidade na execução dos algoritmos de aprendizado e, consequentemente, no processo de classificação.

3.1 Estudo do Problema

Vários problemas requerem o aprendizado de uma função de classificação apropriada. A função atribui um dado padrão de entrada a uma das finitas classes do problema. A escolha das características, atributos, ou medidas usadas para representar os padrões que serão apresentados ao classificador afetam, dentre outros aspectos:

- A precisão da função de classificação;
- O tempo necessário para aprender uma função de classificação;
- O número de exemplos necessário para se aprender uma função de classificação;
- O custo de se executar a classificação usando a função de discriminação aprendida;
- A compreensibilidade do conhecimento adquirido através do treinamento.

3.2 Aplicação em Problemas Reais

Uma das aplicações mais importantes do uso de seleção de características é a utilização da técnica com objetivo de se pré-processar os dados que serão usados em um processo de mineração de dados (*datamining*). Existem, entretanto, muitas outras aplicações.

Em problemas reais, geralmente encontram-se dificuldades relacionadas aos dados coletados. Entre essas dificuldades, é possível citar o elevado número de características, os atributos isolados, que não descrevem informações suficientes do padrão apresentado, e a alta dependência entre as características individuais.

Seres humanos são ineficientes em formular e entender hipóteses quando os dados coletados possuem um grande número de variáveis. Problemas demográficos, análise de dados biológicos ou classificação e categorização de textos são alguns exemplos que apresentam grandes dificuldades de resolução devido ao grande número de variáveis neles presentes. Esses mesmos problemas se tornam facilmente analisados se o espaço de representação for reduzido. A seleção de características procura reduzir a dimensionalidade, priorizando as informações mais relevantes, permitindo, dessa forma, que os algoritmos de mineração de dados trabalhem de forma eficiente. Algumas aplicações ilustrativas do uso do processo de seleção de características são apresentadas.

3.2.1 Categorização de Texto

Categorização, ou agrupamento, de textos é o problema de se atribuir categorias, ou classes, pré-definidas a documentos, com o objetivo de organizá-los [45], agregando documentos similares, de maneira a melhor discriminar documentos pertencentes a grupos diferentes. O problema é de grande importância, dado o enorme volume de documentos disponíveis na Web, e-mails, bancos de dados corporativos, registros médicos, livrarias virtuais, dentre outros. Segundo NIGAM *et al.* [46], a maior dificuldade na categorização está relacionada à alta dimensionalidade do espaço de características. O espaço original consiste de termos isolados (palavras e/ou frases) com altos valores numéricos até para textos de médio tamanho. Isso torna o problema proibitivo para vários algoritmos. Assim, busca-se a redução do espaço sem, no entanto, sacrificar a precisão da categorização. YANG e PEDERSEN [47] utilizaram diferentes métodos e os compararam, com o objetivo de reduzir o espaço de características em problemas de categorização. Os resultados mostraram que os algoritmos são capazes de remover de 50% a 90% dos termos (características) sem uma redução significativa da precisão de classificação.

3.2.2 Recuperação de Imagem

A seleção de características em uma imagem tem grande importância para sistemas de visão computacional e pode ser aplicada à recuperação de imagens, [48], com base no conteúdo. Recuperação com base no conteúdo, [49], é uma técnica proposta para manipular o grande número de imagens hoje existentes. Uma grande quantidade de informações presentes em uma imagem, como por exemplo, a cor, a textura e a forma, podem ser ou não relevantes para o processo de segmentação, classificação ou análise. Ao invés das imagens serem classificadas por nomes, como acontece na categorização de textos, elas são indexadas pelo seu conteúdo visual (características). O grande problema em se fazer recuperação de imagens com base no conteúdo para grandes bases de dados é a "maldição da dimensionalidade" (*curse of dimensionality*), [50], que, dado um conjunto de treinamento, a classificação só melhora até um determinado número ótimo de dimensões, a partir do qual, o desempenho do classificador piora. Como citado por RUI *et al.* [49], a dimensão do espaço de características para esse problema é, em geral, de ordem 10^3. A redução da dimensionalidade é uma alternativa para a solução desse problema. As imagens, após o processo, são indexadas pelas principais características que as constituem.

3.2.3 Detecção de Intrusos

Redes de computadores desempenham um papel fundamental na sociedade moderna e, por isso, se tornaram alvos de invasores, intrusos e criminosos. A segurança de um computador é comprometida quando um intruso o invade. A detecção de intrusos é uma maneira de proteger computadores contra invasões. LEE *et al.* [51] sugerem um *framework* de mineração de dados para análise e construção de um modelo de detecção de intrusos. Os modelos analisam, primeiramente, a movimentação de informações em uma base de dados específica para o problema trabalhado. Dessa análise surgem certos padrões de comportamento dos usuários considerados normais. A seleção de características atua nessa etapa, já que alguns padrões são descritos por várias características. O processo seleciona as características que mais determinam, sem perda de precisão, o padrão do indivíduo. O padrão de um novo usuário, ou de um visitante não cadastrado, é então comparado com o dos usuários normais e, de maneira geral, se houver muita diferença entre ambos os padrões de comportamento comparados, o usuário é identificado como intruso e atitudes apropriadas podem, então, ser tomadas.

3.2.4 Análise de Expressão Gênica

Dados funcionais e estruturais provenientes da análise do genoma humano cresceram exponencialmente e, com isso, trouxeram grandes oportunidades e desafios para a mineração de dados. Em particular, a expressão genética em *microarrays* (microarranjos) é uma tecnologia crescente que permite a visualização da expressão de centenas ou milhares de genes em um único experimento, [52]. No entanto, o número de amostras desses experimentos é extremamente reduzido. DOAK [53], por exemplo, apresenta casos com algumas poucas dezenas de exemplos de treinamento, mas com algumas milhares de características. Em problemas com grande dimensionalidade, como é o caso dos problemas com análises genômicas, têm-se muitos atributos irrelevantes, o que ocasiona em aumento de complexidade computacional e perda de exatidão na tarefa de classificação. Nesses casos, torna-se necessária a remoção de características irrelevantes e a definição de um subconjunto, menor, de características discriminativas para melhorias na classificação, [54].

3.3 Graus de Relevância

Determinar quais características são relevantes à tarefa de aprendizado é uma grande preocupação dos que lidam com aprendizado de máquina. Isso se deve ao fato de que a inclusão de uma característica irrelevante, ou redundante, pode ocasionar uma redução drástica no desempenho dos algoritmos.

Para determinar se uma característica específica é ou não relevante ao processo de aprendizado e classificação, é preciso compreender os conceitos de relevância. Existem várias definições na literatura de aprendizado de máquina sobre o significado de uma característica ser "relevante". JOHN *et al.* [55] e GUYON [56] definem duas notações para relevância:

- Relevância forte: um atributo f_i é considerado fortemente relevante se a distribuição de probabilidade dos valores das classes, dado o conjunto completo de características F, modificar-se quando f_i for removido, ou seja, a remoção do atributo causa uma degradação no classificador;

- Relevância fraca: um atributo f_i é considerado fracamente relevante se ele não for fortemente relevante e existir um subconjunto de variáveis $F'(F' \subset F)$ em que o desempenho do classificador usando $F' \cup \{f_i\}$ é superior ao desempenho do mesmo classificador utilizado somente sobre subconjunto F'.

Por sua vez, YU e LIU [57] dividem as características fracamente relevantes em redundantes e não redundantes. Uma característica é redundante em relação a outra se seus valores estão correlacionados.

Existem, ainda, características que não possuem relevância forte e nem fraca e, por isso, são denominadas de irrelevantes. As características redundantes e irrelevantes não devem ser selecionadas.

3.4 Características Gerais

Uma forma conveniente para representar as abordagens para a seleção de atributos é a busca heurística, na qual cada estado no espaço de busca é composto por um subconjunto de possíveis atributos. A Figura 3.1 mostra subconjuntos possíveis para quatro atributos: círculos brancos indicam que o atributo em questão não foi selecionado; círculos pretos indicam a seleção do atributo. Cada estado no domínio de subconjuntos de atributos especifica quais atributos foram selecionados. Nota-se que os estados são parcialmente ordenados, sendo que os estados à direita acrescentam um atributo aos da esquerda.

Desse modo, qualquer método de seleção de atributos pode ser caracterizado por sua instanciação em relação a quatro questões básicas, as quais determinam a natureza do processo de busca heurística, [58]:

- Ponto de partida: Dependendo do ponto de partida escolhido, a direção de busca irá variar. Quando a busca tem seu estado inicial mais à esquerda, o conjunto vazio de atributos, representado pelos quatro círculos em brancos, ela é conhecida como busca "para frente" (*forward selection*). Já a situação que

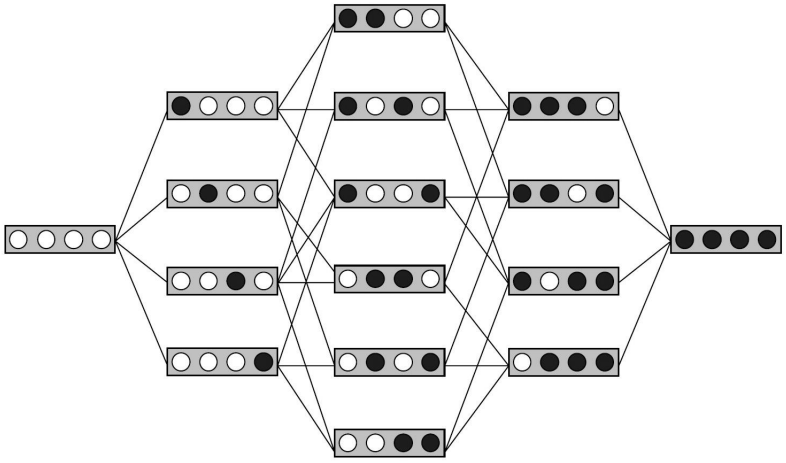


Figura 3.1: Subconjuntos possíveis para quatro atributos.

inicia com o subconjunto contendo todos os atributos e vai, sucessivamente, removendo-os, é denominada de eliminação (ou sentido da busca) "para trás" (*backward selection*). Também podem ser empregadas variações de ambas as técnicas, selecionando-se um estado inicial em algum ponto do espaço de busca e movendo-se a partir desse ponto (*outward selection*);

- Organização da busca: A cada ponto na busca, modificações locais no conjunto de atributos são consideradas; uma dessas é selecionada e uma nova iteração é realizada. Claramente, se o número de características é muito grande, a busca exaustiva de todos os subespaços é proibitiva, já que existem (2^n) subconjuntos diferentes para n características (considerando também o subconjunto vazio, sem nenhuma característica). Uma abordagem mais prática é a utilização de uma busca heurística, porém não garante uma solução ótima;

- Critério de avaliação: A estratégia utilizada na avaliação dos subconjuntos de características é um problema importante. Uma métrica normalmente empregada envolve a habilidade de um atributo discriminar as classes de um conjunto de treinamento. Para classificação, o melhor subconjunto é aquele que provê a maior separação dos dados. A separação de dados é normalmente calculada por algum critério de medida da distância entre as classes existentes. Define-se precisão de classificação como sendo a porcentagem de exemplos classificados corretamente para um determinado conjunto de teste;

- Critério de parada: Durante o processo de avaliação, deseja-se que o algoritmo pare quando é observado que já não existem melhorias na precisão do classificador.

De maneira geral, o processo geral dos algoritmos de seleção de características é estruturado nos passos descritos na Figura 3.2.

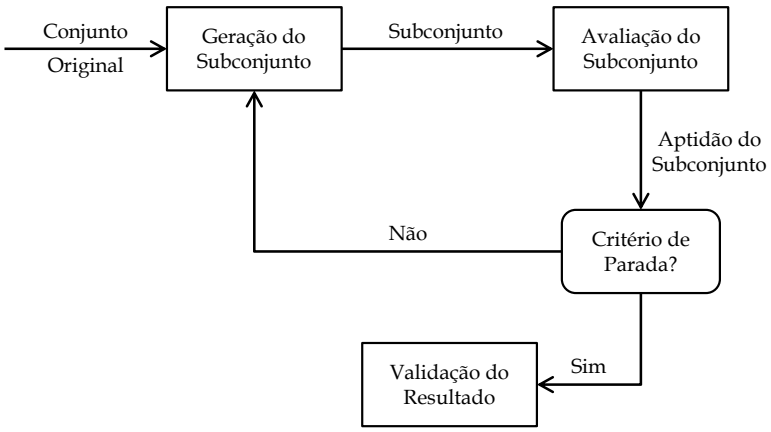


Figura 3.2: Estrutura dos algoritmos de seleção de características.

SIEDLECKI e SKLANSKY [59], DASH e LIU [60] e JAIN e ZONGKER [61] criaram diferentes categorias para comparar e avaliar os diferentes métodos de seleção de características. Algumas avaliações dividem os métodos de seleção considerando o procedimento de busca e a função de avaliação dos mesmos. Os procedimentos de busca são divididos em busca completa, heurística ou aleatória e as funções de avaliação estão divididas em funções, ou medidas, relacionadas à distância, informação, dependência, consistência e taxa de erro de classificação. As categorias encontradas em GUYON e ELISSEEFF [62] são descritas nas seções a seguir.

3.5 Métodos de Seleção em Filtro

Técnicas de seleção de atributos em filtro introduzem um processo separado, o qual ocorre antes da aplicação do algoritmo de aprendizado. O modelo foi batizado de filtro por JOHN *et al.* [55] pelo fato de "filtrar" os atributos irrelevantes, segundo algum critério, antes que uma indução ocorra, ou seja, as características são selecionadas antes que o algoritmo de aprendizado seja executado. Esse passo de pré-processamento considera características gerais do conjunto de exemplos para selecionar alguns atributos e excluir outros. Sendo assim, métodos de filtros são independentes do algoritmo de classificação que, simplesmente, receberá como entrada o conjunto de exemplos contendo apenas os atributos selecionados pelo filtro. A vantagem do modelo em filtro está no fato de que o mesmo não precisa ser reaplicado para cada execução do algoritmo de treinamento. Assim, o modelo em filtro é eficiente ao se abordar problemas que possuam um espaço de características muito elevado.

Qualquer algoritmo que efetue algum tipo de seleção pode ser usado para filtrar atributos. A saída do algoritmo de filtragem é o conjunto de atributos por ele selecionados. Os atributos restantes são removidos do conjunto de exemplos, reduzindo

assim sua dimensão. Após isso, o conjunto de exemplos reduzido pode ser usado por qualquer classificador. Contudo, sabe-se, que na maioria das vezes, o subconjunto ótimo de características depende do algoritmo de treinamento a ele associado. Portanto, embora a abordagem apresente-se computacionalmente eficiente, ela encontra apenas soluções sub-ótimas genéricas que independem do classificador. Assim, um subconjunto de características selecionado usando um método em filtro pode resultar em uma alta precisão para determinado classificador e em baixa precisão em outros. O maior problema destes métodos é que cada coeficiente é computado utilizando somente informações do atributo relacionado, não levando em conta a existência de uma interdependência, ou mútua informação, entre as características. De fato, podem existir atributos complementares que individualmente não têm uma relevância, mas que combinados podem ter um papel importante no processo de discriminação. Dentre os métodos de filtro, pode-se citar o Gulob, o Fisher e o Relief.

3.5.1 Golub e Fisher

Esses métodos se baseiam na diferença da magnitude dos níveis de expressão de cada atributo. Um atributo pode ser dito diferencialmente expresso em duas classes distintas se a diferença da média ponderada dessas classes dividida pela soma do seu desvio padrão for elevada. Definindo μ_1 e μ_2 como as médias das duas classes, e σ_1 e σ_2 como seus desvios padrões, respectivamente, pode-se definir as pontuações (*scores*) Golub G, [63], e Fisher F, [64], da seguinte forma:

$$G = \frac{|\mu_1 - \mu_2|}{(\sigma_1 + \sigma_2)} \tag{3.1}$$

$$F = \frac{(\mu_1 - \mu_2)^2}{(\sigma_1^2 + \sigma_2^2)} \tag{3.2}$$

3.5.2 Relief

O algoritmo Relief, [65], trabalha por meio da amostragem aleatória de exemplos do conjunto de dados e localização do vizinho mais próximo da mesma classe e do vizinho mais próximo da classe oposta. Os valores dos atributos dos vizinhos mais próximos são comparados aos da classe amostrada e utilizados para atualizar os pesos de relevância de cada atributo em relação à classe. A idéia do método é que atributos importantes devem diferenciar exemplos de classes diferentes e possuir valores similares para exemplos da mesma classe.

3.6 Métodos de Seleção Embutidos

A estratégia dos métodos embutidos, também chamada de seleção em cápsula, se baseia no fato de que alguns indutores são capazes de realizar sua própria seleção de atributos. Eles fazem uso do algoritmo de indução para estimar o valor do subconjunto de características selecionado durante a fase de treinamento e são geralmente específicos para um dado algoritmo de classificação. A idéia central desses métodos se baseia na otimização direta de uma função objetivo composta geralmente de dois termos. O primeiro, relacionado a uma medida do desempenho do classificador, que deve ser maximizada, e o segundo, uma medida de regularização relacionada à quantidade de variáveis, que deve ser minimizada. Dentre os métodos embutidos, pode-se citar o AROM e o NSC. Outro exemplo de método embutido é fazer a utilização do algoritmo IMA$_p$ na sua formulação L_∞, que minimiza a norma L_1 do vetor w. Nessa versão, ele seleciona as características pelo valor da maior componente do vetor, minimizando as componentes não importantes, sendo essas, assim, facilmente eliminadas.

3.6.1 *Approximation of the Zero-Norm Minimization*

Esse método, proposto por WESTON *et al.* [66], utiliza a noção de norma-0 do vetor w, normal ao hiperplano separador de classificadores lineares, para selecionar conjuntos de características. A idéia é limitar o número de componentes não-nulas e diminuir a norma de w, mantendo o erro de treinamento pequeno. Dessa forma, o número de características utilizadas e o erro de generalização do classificador podem ser minimizados simultaneamente. Esse processo é chamado de Aproximação de Minimização de Norma Zero (AROM – *Approximation of the Zero-Norm Minimization*), pois o número de componentes não nulas é minimizado. Intuitivamente, o AROM altera a influência das características gradativamente, reescalando-as em cada iteração. Assim, as características que não são necessárias para classificar corretamente os exemplos tendem rapidamente a assumir valores muito próximos a zero. Por outro lado, as características relevantes tendem a ganhar mais importância. Ao final do processo, apenas as características relevantes terão valores diferente de zero, podendo, assim, eliminar as demais do processo de classificação.

3.6.2 *Nearest Srhunken Centroid*

O método *Nearest Srhunken Centroid* (NSC), desenvolvido por TIBSHIRANI *et al.* [67], baseia-se no classificador do protótipo mais próximo. Consiste em atribuir a uma nova amostra a classe cujo protótipo (neste caso, o centróide) esteja mais próximo. O método usa centróides "encolhidos" como protótipos, resultantes da

identificação de subconjuntos de atributos que melhor caracterizam cada classe. O algoritmo tenta reduzir os protótipos da classe (μ_{C_j}) para a média global:

$$\mu = \frac{1}{m} \sum_i x_i \tag{3.3}$$

Resumidamente, o algoritmo calcula:

$$d_j = \frac{\mu_{C_j} - \mu}{m_j\,(s)}, \tag{3.4}$$

onde $m_j = \sqrt{1/|C_j| - 1/m}$ e s é um vetor de variâncias agrupadas dentro da classe para cada função. Pode-se agora ver o centróide da classe como:

$$\mu_{C_j} = \mu + m_j\,(s \cdot d_j) \tag{3.5}$$

Diminuindo o valor de d_j, pode-se mover o centróide da classe para o centróide geral. Quando uma componente do centróide da classe é igual à componente correspondente da média geral das classes, o atributo já não desempenha um papel na classificação e pode efetivamente ser removido. O método "encolhe" progressivamente o valor de d_j, obtendo novos valores $d_j^{'}$:

$$d_j^{'} = \text{sinal}\,(d_j) \cdot (|d_j| - \Delta)\,, \tag{3.6}$$

onde Δ (*threshold*) é um valor positivo. Portanto, como d_j diminui progressivamente, mais atributos vão sendo removidos.

3.7 Métodos de Seleção *Wrapper*

Em contraste com filtros, a abordagem *wrapper*, popularizada por KOHAVI e JOHN [68], gera vários subconjuntos de atributos como candidatos, executa o indutor individualmente em cada subconjunto e usa a precisão do classificador para avaliar o subconjunto em questão. O processo é repetido até que um critério de parada seja satisfeito. A idéia geral dessa abordagem é que o algoritmo de seleção de características existe como um *wrapper* ao redor do indutor e é responsável por conduzir a busca por um bom subconjunto de atributos. A qualidade de um subconjunto candidato é avaliada pelo próprio indutor como uma "caixa preta". O objetivo da busca é encontrar o subconjunto (nó) com a melhor qualidade, utilizando alguma função para guiá-la.

Em geral, a busca é conduzida no espaço do subconjunto de atributos, ou de candidatos. Como estratégia pode-se empregar algoritmos míopes (*greedy*) ou bus-

cas direcionadas (*best-first*) e com direções *forward* ou *backward*. A precisão dos subconjuntos pode ser estimada por uma validação cruzada (*cross-validation*). A principal vantagem do modelo é a dependência entre os algoritmos de seleção e de aprendizado. Por outro lado, essa abordagem pode ser computacionalmente dispendiosa, uma vez que o indutor deve ser executado para cada subconjunto de atributos considerado. Dentre os métodos, pode-se citar o RFE e o AOS, método proposto por esse trabalho e apresentado no Capítulo 4.

3.7.1 *Recursive Feature Elimination*

A idéia básica do método da eliminação recursiva de características, denominado de *Recursive Feature Elimination* (RFE) e introduzido por GUYON *et al.* [69], é a eliminação recursiva da menor componente do vetor w, uma vez que ela não tem muita influência sobre a posição do hiperplano. A cada passo do processo, um número fixo de componentes é eliminado e o classificador é retreinado. A eliminação recursiva de uma característica por vez gera um classificador com menos erros esperados, quando comparada à remoção de mais de uma característica ao mesmo tempo. Uma consideração a ser feita em relação ao método RFE é que, se a margem geométrica do vetor w for definida por:

$$\gamma_g = \frac{\gamma_g}{||w||} w, \tag{3.7}$$

pode-se observar que a magnitude das componentes $|w_j|$ do vetor tem uma relação direta com a magnitude da margem projetada do vetor em um subespaço no qual a j-ésima característica é excluída, como é dada pela equação:

$$\gamma_{p_j}^{d-1} = \frac{\gamma_g^d}{||w||} \left(\sum_{k \neq j} w_k^2 \right)^{1/2}, \tag{3.8}$$

na qual $\gamma_{p_j}^{d-1}$ indica a magnitude da margem geométrica quando removida a característica j em um espaço de dimensão d. Por razões de simplicidade, chama-se esse valor de margem projetada.

É de fácil constatação que a remoção da menor componente relacionada com a menor margem projetada não resultará sempre na melhor margem geométrica para um problema de dimensão $d - 1$. Nesse sentido, se estabelece uma relação direta, usando a Minimização do Risco Estrutural, entre o valor da margem e a capacidade de generalização do classificador, e percebe-se que o RFE nem sempre escolhe a característica adequada para remover. Assim, o uso de uma estratégia míope de exploração dos espaços possíveis não garante sempre um classificador ótimo associado ao processo de seleção de características.

Capítulo 4

Admissible Ordered Search

Complex theories do not work;
simple algorithms do.

Vladimir Vapnik

Técnicas de seleção de características comumente utilizadas com base em análises de variância dos dados, bem como métodos baseados na eliminação recursiva de características, nem sempre encontram classificadores com uma menor quantidade de atributos ou um melhor poder de generalização. Nesse contexto, foi introduzido um novo algoritmo de seleção de características, [70], denominado *Admissible Ordered Search* (AOS), o qual se baseia na realização de uma busca ordenada admissível, semelhante à empregada pelo algoritmo A*, [71]. Esse algoritmo tem a capacidade de encontrar, em cada dimensão do problema, o classificador de maior margem.

Em um processo de busca ordenada, garante-se a admissibilidade do algoritmo se a função de avaliação é monótona. Para problemas de minimização, a função precisa ser monótona crescente, e, para problemas de maximização, ela deve ser monótona decrescente. Nesse sentido, uma vez que se está à procura da maximização da margem, usa-se como função de avaliação o valor da margem obtido a partir de cada hipótese após a solução do problema de classificação no conjunto de características selecionado por essa hipótese. A admissibilidade do processo é garantida, uma vez que os valores da margem são sempre decrescentes quando a dimensão diminui. Ou seja:

$$\gamma_{g_j}^{d-1} \leq \gamma_g^d, \forall j \tag{4.1}$$

A estratégia de controle dessa busca ordenada é implementada com a inserção das hipóteses candidatas em uma fila ordenada pelos valores das margens. Uma vez que a ordem das características selecionadas não importa, poderá haver alguma redundância. A fim de evitar esse problema, cria-se uma tabela *hash*, e, para cada nova hipótese, verifica-se a sua unicidade nessa tabela antes de inseri-la na fila. Usando a

margem geométrica real exigiria a solução de um problema para a maximização da margem para cada hipótese gerada. Em vez disso, utiliza-se uma estimativa otimista desta margem, que é a sua margem projetada. Esse valor será um limite máximo para a margem geométrica real da mesma hipótese no espaço associado. Com isso, mantém-se a admissibilidade do processo, uma vez que:

$$\gamma_{p_j}^{d-1} \geq \gamma_{g_j}^{d-1}, \forall j \tag{4.2}$$

$$\gamma_{p_j}^{d-1} \leq \gamma_{g}^{d}, \forall j \tag{4.3}$$

Para cada iteração do algoritmo, a hipótese relacionada com a maior margem é escolhida, independentemente da sua dimensão, para ser expandida e gerar novas hipóteses num espaço de dimensão menor. Dessa forma, pode-se verificar duas situações possíveis:

- Primeiro: para os casos em que o valor da margem para a hipótese escolhida é o valor projetado, pode-se calcular o seu valor real através da solução de um problema de maximização da margem e compará-lo com o maior valor da fila de prioridade. Se ainda é a melhor opção, fecha-se esse estado e geram-se as suas hipóteses. Caso contrário, substitui-se o valor da margem projetada dessa hipótese pelo valor real e reinsere o mesmo na fila;

- Segundo: para os casos em que o valor da margem da hipótese escolhida já é o valor real, fecha-se o estado e geram-se suas hipóteses.

Para amenizar a explosão combinatória, o algoritmo desenvolve dois esquemas de podas adaptativas, sendo um baseado na atualização constante de um limite de margem inferior. Esse limite é computado toda vez que uma hipótese escolhida for a primeira a chegar em uma dada dimensão. Para tanto, é utilizada uma estratégia de seleção míope que avalia os valores de margem até uma dimensão inferior escolhida, eliminando, assim, estados de maior dimensão que tenham valores inferiores a esse limite. A outra poda, também realizada quando a hipótese escolhida é a primeira a chegar a uma dada dimensão, é baseada em um corte que elimina todos os candidatos presentes na fila que tenham dimensão superior à hipótese em questão, além do valor do corte. Com isso, são eliminadas hipóteses que já estão há algum tempo na fila e ainda não foram selecionadas, ou seja, hipóteses que tendem a ter valores de margem relativamente baixos. Também, nesse momento, uma estimativa do erro esperado pode ser calculada e utilizada como critério de parada do algoritmo, uma vez que representa o desempenho de generalização do classificador.

4.1 Ramificação

Como primeiro parâmetro de ramificação, adota-se a eliminação das componentes relacionadas aos menores valores de margens projetadas, ou as menores componentes do vetor w, tal como utilizado no RFE. Esse critério está relacionado à escolha da menor variação no valor da função objetivo do problema SVM de minimização em sua formulação dual:

$$J = 1/2 \cdot \alpha^T \cdot H \cdot \alpha - \alpha^T \cdot 1, \tag{4.4}$$

sendo α o vetor de multiplicadores e H a matriz definida a partir da matriz *kernel* K com componentes na forma $y_i \cdot y_j \cdot K_{i,j}$. Assim, sustentando o mesmo vetor de multiplicadores, se for retirada a i-ésima variável, tem-se a variação:

$$\begin{aligned} \Delta J(i) &= J - J(i) \\ &= 1/2 \cdot \alpha^T \cdot H \cdot \alpha - 1/2 \cdot \alpha^T \cdot H(i) \cdot \alpha \\ &= 1/2 \cdot \alpha^T \cdot (H - H(i)) \cdot \alpha \end{aligned} \tag{4.5}$$

Dessa forma, escolhe-se como variável a ser retirada aquela que produzir a menor variação no valor da função, ou seja:

$$k = \text{Arg Min}_i \Delta J(i) \tag{4.6}$$

Para o caso especial de SVM linear na formulação dual, tem-se o valor de cada componente da matriz *kernel* K definido como o produto interno dos vetores respectivos, ou seja:

$$K_{i,j} = \langle x_i, x_j \rangle, \tag{4.7}$$

reduzindo o cálculo de cada componente da matriz diferença $H - H(i)$ ao simples produto das componentes associadas a variável que está sendo retirada, ou seja $(x_i)_k \cdot (x_j)_k$.

Na formulação primal, pode-se associar a variação do valor da função à variação da norma do vetor w, ou seja:

$$\begin{aligned} \Delta J(i) &= J - J(i) \\ &= 1/2 \cdot \alpha^T \cdot H \cdot \alpha - 1/2 \cdot \alpha^T \cdot H(i) \cdot \alpha \\ &= ||w||_2 - ||w_i||_2 \end{aligned} \tag{4.8}$$

O que se torna equivalente, em termos de critério, à escolha da variável associada a componente de menor magnitude do vetor. Ou seja:

$$k = \text{Arg Min}_i (w_i)^2 \tag{4.9}$$

Esse critério de escolha é adotado pelo algoritmo RFE quando combinado com o SVM linear na formulação primal.

Como segundo critério de ramificação utiliza-se uma medida relacionada à minimização da expectativa de um limite superior associado ao erro esperado do processo de classificação. Segundo VAPNIK [9], esse limite pode ser expresso por:

$$\frac{1}{\ell} \cdot E\left\{\frac{R^2}{\gamma^2}\right\}, \tag{4.10}$$

onde ℓ se refere à quantidade de amostras, R se refere ao tamanho dos dados relacionado ao raio da maior esfera que engloba os mesmos e γ se refere à margem do classificador. Ou seja, o desempenho dos diferentes candidatos em termos de poder de generalização depende não somente de uma larga margem, mas também de uma redução no tamanho dos dados.

Portanto, se o objetivo é minimizar a relação R^2/γ^2 ou, de forma equivalente, minimizar o valor da expressão $R^2 \cdot ||w||_2^2 = R^2 \cdot \sum_i (w_i)^2$, deve-se retirar a variável, ou característica, que menos altera a diferença desse valor, no sentido de preservar aquelas que mais alteram sua diferença. Ou seja:

$$k = \text{Arg Min}_i (R_i)^2 \cdot (w_i)^2 \tag{4.11}$$

Como terceiro critério de ramificação utiliza-se uma medida relacionada à maximização da distância entre os centros das duas classes. Isto é, o desempenho dos diferentes candidatos em termos de poder de generalização depende não somente de uma larga margem, mas também de uma mínima redução na distância entre os centros das classes.

Portanto, se o objetivo é minimizar o valor da expressão $(1/C)^2 \cdot \sum_i (w_i)^2$, na qual C é a distância entre os centros, deve-se retirar a variável que menos altera a diferença desse valor. Logo:

$$k = \text{Arg Min}_i \frac{(w_i)^2}{(C_i)^2} \tag{4.12}$$

Como quarto critério, utiliza-se a junção dos dois últimos; uma medida relacionada à minimização do raio juntamente com a maximização da distância entre os centros. Portanto, o desempenho dos diferentes candidatos em termos de poder de generalização depende de uma mínima redução na margem e na distância entre os centros, além de uma maior redução no tamanho dos dados. Ou seja:

$$k = \text{Arg Min}_i \frac{(R_i)^2 \cdot (w_i)^2}{(C_i)^2} \tag{4.13}$$

Adicionalmente, é utilizado um critério associando o conceito dos métodos em filtro. Para isso, utilizou-se a pontuação do método Golub. Assim, o desempenho dos diferentes candidatos em termos de poder de generalização depende, além de uma larga margem, de um mínimo *score* G. Ou seja:

$$k = \text{Arg Min}_i \left(\frac{|\mu_{1_i} - \mu_{2_i}|}{(\sigma_{1_i} + \sigma_{2_i})} \right)^2 \cdot (w_i)^2 \text{ ou } (G_i)^2 \cdot (w_i)^2 \tag{4.14}$$

4.2 Medidas de Avaliação

Além da medida de avaliação padrão, o valor da margem, foi adotada outra medida, sendo esta a distância entre os centros das classes, que também é uma medida monótona decrescente, uma vez que a distância em uma dimensão $d-1$ é sempre menor ou igual à distância na dimensão d.

Como, para o caso de um classificador SVM, deve haver o comprometimento com a margem, essa medida foi utilizada em conjunto com o valor da margem. Então, como segunda medida de avaliação, tem-se $\gamma \cdot C$, sendo C a distância entre os centros das duas classes.

4.3 Solução Inicial

Como foi visto, a escolha da variável associada à componente de menor magnitude do vetor w está relacionada à escolha da margem projetada de maior valor. De fato, se for escolhida uma variável associada a uma componente de valor zero, tem-se o valor da margem projetada igual ao valor da margem máxima real no subespaço associado. Isso facilita bastante o processo de solução dos problemas de classificação com SVM, pois, nesse caso, não há necessidade de se resolver o novo problema de classificação, uma vez que as soluções são as mesmas.

Quando o valor da componente de menor magnitude for muito baixo, tem-se a solução dos dois problemas muito próximas. Nesse caso, utiliza-se a última solução obtida no espaço em questão como solução inicial do problema de classificação do subespaço inferior associado. Se a formulação for primal, elimina-se a componente associada do vetor w. Entretanto, caso a formulação seja dual utiliza-se o mesmo vetor de multiplicadores α.

Para o cálculo da margem projetada na formulação dual utiliza-se a mesma relação de normas do vetor w computados a partir de sua expansão em função do vetor de multiplicadores α. Ou seja:

$$\gamma_{p_j}^{d-1} = \left(\frac{1}{||w||_2} \right) \cdot \gamma_g^d \cdot ||w_j||_2 = \left(\frac{1}{\alpha^T \cdot H \cdot \alpha} \right) \cdot \gamma_g^d \cdot \alpha^T \cdot H\,(j) \cdot \alpha \tag{4.15}$$

4.4 Pseudocódigo do Algoritmo AOS

O Algoritmo 4.1 descreve o pseudocódigo do algoritmo AOS.

Algoritmo 4.1: *Admissible Ordered Search*

Entrada: conjunto de treinamento: Z;
conjunto de características: $F = \{1, 2, 3, \ldots, d\}$;
fator de ramificação: b;
profundidade da poda: p;
profundidade do corte: c;
nível de parada: s;
Dados: nível alcançado: *nivel*;
limite inferior usado para podar a árvore de busca: *limite*;
Saída: último estado aberto;

```
1  início
2      inicializar o heap H e a tabela hash HT;
3      computar a solução SVM para o estado inicial S_inicial com o conjunto F;
4      nivel ← d;
5      inserir S_inicial em H;
6      enquanto nivel > s e H não vazio faça
7          selecionar a melhor hipótese S de H;
8          se S tiver somente uma margem projetada então
9              computar a margem real de S usando SVM;
10             se solução SVM convergiu então
11                 reinserir S em H;
12             fim se
13         senão
14             se dimensão do estado S for igual a nivel então
15                 usar o RFE até nivel − p e encontrar o novo valor para limite;
16                 eliminar de H todos os estados com margem menor que limite;
17                 eliminar de H todos os estados com nível maior que nivel + c;
18                 nivel ← nivel − 1;
19             fim se
20             ordenar F em S pelo critério de ramificação;
21             para i de 1 até b faça
22                 criar um novo estado S' com o conjunto F' = F − {f_i};
23                 computar γ_pj para S'; // onde γ_pj é a margem projetada
24                 se γ_pj > limite e F' não está em HT então
25                     inserir F' em HT;
26                     inserir S' em H;
27                 fim se
28             fim para
29         fim se
30     fim enqto
31     retorna último estado aberto;
32 fim
```

Capítulo 5

Experimentos e Resultados

If you torture the data for long
enough, in the end they will confess.

Ronald H. Coase

5.1 Conjuntos de Dados

Para a análise dos resultados, foram utilizadas dezesseis bases de dados. Onze bases linearmente separáveis, sendo duas sintéticas, cinco de *microarrays*, uma gerada artificialmente e outras três "comuns", e cinco não linearmente separáveis. Com exceção da base artificial, as bases utilizadas no trabalho estão contidas no repositório de aprendizado de máquinas da UCI, [72], ou referenciadas por GOLUB *et al.* [63], ALON *et al.* [73] ou SINGH *et al.* [74].

5.1.1 Bases Lineares

A base de dados Iris é talvez o banco de dados mais utilizado na literatura para aprendizado de máquinas. Ela contém 3 séries de 50 instâncias da planta Íris, da família das Iridáceas, sendo cada conjunto correspondente a uma das três classes Iris-Setosa, Iris-Versicolour e Iris-Virginica. Cada registro é composto por 4 atributos (largura e comprimento da pétala e largura e comprimento da sépala). A combinação da classe Setosa contra as outras 2 deixa a base linearmente separável.

A base Mushroom contém atributos de diversas variedades de cogumelos e a tarefa do classificador é determinar quais são comestíveis e quais não. Ela conta com 98 componentes e possui 5644 amostras.

O objetivo da base Sonar é classificar sinais sonares refletidos de um cilindro de metal ou de um cilindro aproximado de rocha. Ela possui 60 componentes e 208 amostras.

A base Synthetic Control possui 600 exemplos de 6 tipos de gráficos de controle gerados sinteticamente. Ela possui 60 componentes e foi adaptada para um problema de classificação binária e linearmente separável.

A base Robot LP4 possui 117 instâncias e 90 atributos, com as classes divididas entre normal versus colisão e obstrução.

A base de dados Leukemia visa classificar pacientes com leucemia por meio de sua expressão genética. Os dados são uma combinação de amostras de treinamento e validação de 47 pacientes com leucemia linfóide aguda e 25 pacientes com leucemia mielóide aguda. As amostras apresentam informações referentes a 7.129 genes.

A base Breast contém biópsias de 24 pacientes de câncer de mama antes dos 4 ciclos de tratamento taxotere (docetaxel). Ela conta com 12.625 genes.

A base Prostate contém perfis de expressão gênica de câncer de próstata, com 12.600 genes, derivados de amostras de 52 pacientes com tumor e 50 com tecidos normais.

A base Colon contém 62 amostras, com 2.000 genes, de pacientes de câncer de cólon, sendo que 40 biópsias são de tumores, e 22 são de partes saudáveis do cólon do mesmo paciente.

A base DLBCL vem de um estudo da expressão gênica de dois linfomas: linfoma difuso de grandes células B e linfoma folicular. Cada amostra contém 5.468 genes. O número de amostras é 77.

A base artificial possui 1600 pontos, sendo 800 de cada classe, e 20 dimensões. A base será explicada na seção de resultados.

5.1.2 Bases Não Lineares

A base Ionosphere descreve dados sobre radares. Bons resultados são considerados se mostram evidência de algum tipo de estrutura na ionosfera, caso contrário são considerados ruins. Esse conjunto de dados é composto por 34 atributos e 351 exemplos.

A base WDBC (Wisconsin Diagnostic Breast Cancer) contém dados de 569 diagnósticos de pacientes e busca identificar a presença de tumores de mama que podem ser malignos ou benignos.

O problema da base Bupa é predizer se um paciente, do sexo masculino, possui ou não disfunção hepática tomando-se como base diversos exames sanguíneos e a quantidade de álcool consumida. Ela possui 345 exemplos com 6 atributos.

O problema da base Pima é predizer se uma paciente, mulher de descendência indígena Pima com idade mínima de 21 anos, seria classificada como diabética, segundo o critério estabelecido pela Organização Mundial de Saúde, fornecidos dados clínicos e laboratoriais. Ela conta com 768 exemplos, cada um com 8 atributos.

A base Wine possui 178 instâncias de vinho. Os dados são resultado de uma análise química realizada em vinhos produzidos numa mesma região da Itália, mas vindos de três diferentes cultivares. A análise determinou as quantidades de 13 constituintes encontrados em cada um dos três tipos de vinhos. As classes foram divididas nos tipos 1 e 3 contra o tipo 2.

5.1.3 Resumo das Bases

A tabela 5.1 mostra um resumo das informações das bases de dados, com as quantidades de características (atributos) e amostras (instâncias) de cada uma.

Tabela 5.1: Informações das bases de dados.

Base	Atributos	Amostras		
		Pos.	Neg.	Total
Iris	4	50	100	150
Mushroom	98	3488	2156	5644
Sonar	60	97	111	208
Synthetic	60	300	300	600
Robot LP4	90	24	93	117
Colon	2000	22	40	62
Leukemia	7129	47	25	72
Prostate	12600	50	52	102
Breast	12625	10	14	24
DLBCL	5468	58	19	77
Artificial	20	800	800	1600
Ionosphere	34	225	126	351
WDBC	30	212	357	569
Bupa	6	145	200	345
Pima	8	268	500	768
Wine	13	107	71	178

5.2 Resultados

5.2.1 IMA$_p$

Para a validação do algoritmo IMA$_p$, [36], foram testadas as formulações IMA$_1$, IMA$_2$ e IMA$_\infty$ nos conjuntos de pontos propostos por PEDROSO e MURATA [75]. A sugestão foi iniciar com os pontos $(0, 1)$ e $(2, 0)$, cada um pertencendo a uma classe diferente, e ir adicionando um ponto aleatório a cada classe. Uma classe é representada pelo símbolo $\times$ e a outra pelo símbolo $+$. A Figura 5.1 apresenta os testes. A linha sólida corresponde ao hiperplano separador referente à formulação IMA$_\infty$, a tracejada ao hiperplano da IMA$_2$ e a pontilhada ao da IMA$_1$. Os resultados

obtidos são os mesmos apresentados pela solução analítica proposta pelos autores. Os valores das normas ($||w||_q$) e das margens (γ) relativas a cada formulação são mostrados na Tabela 5.2.

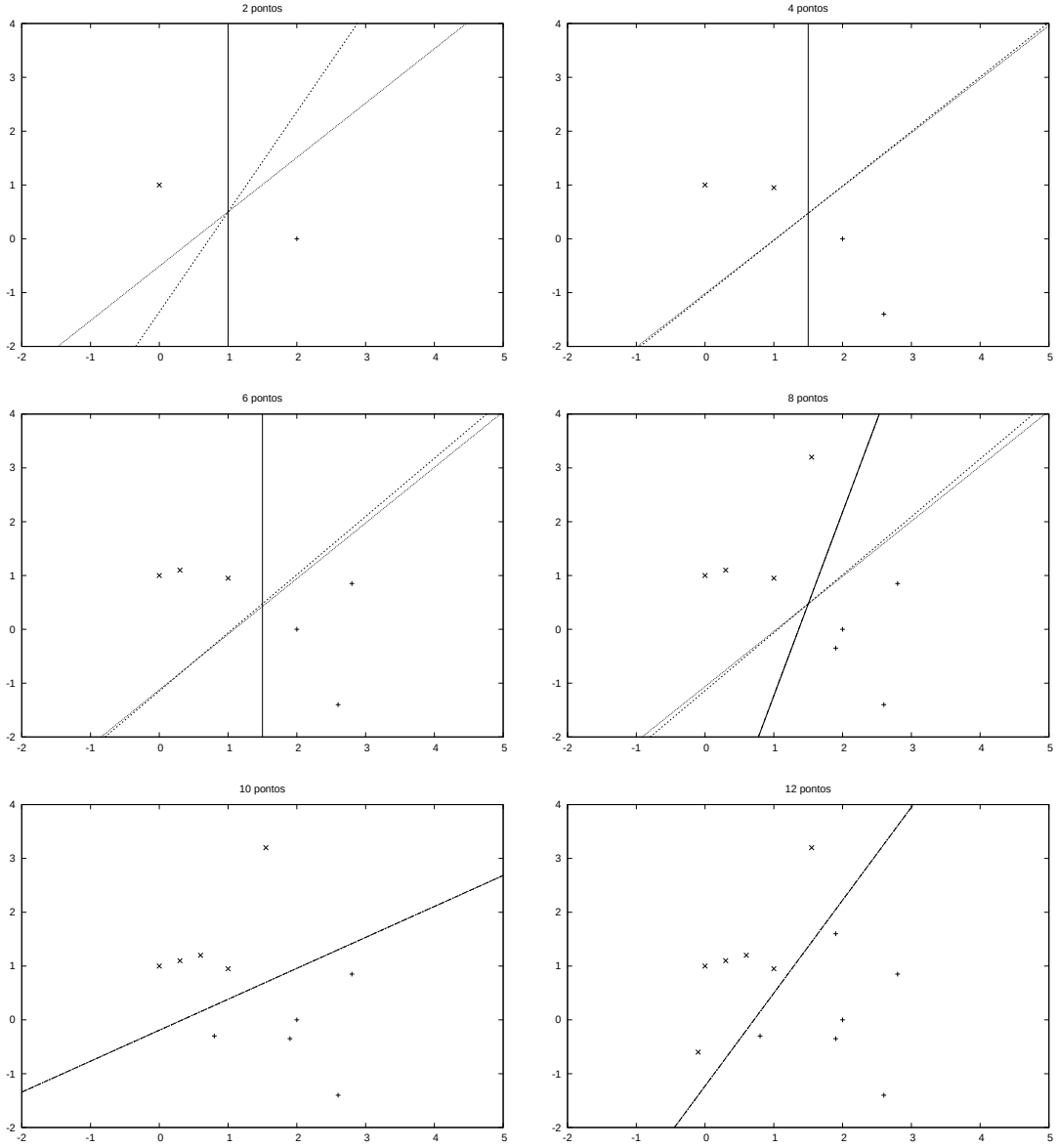


Figura 5.1: Diferentes formulações arbitrárias para diferentes conjuntos de pontos.

Ainda para demonstrar a utilização do algoritmo IMA$_p$, foram realizados testes com as formulações IMA$_1$, IMA$_2$ e IMA$_\infty$ na solução de dois problemas, Mushroom e Sonar, de separabilidade linear e os resultados foram comparados com os obtidos pela solução do algoritmo SMO. A comparação entre o IMA$_p$ e o SMO foi feita pela norma L_2, que calcula o valor da margem (γ) pelo valor da distância euclidiana, ou seja, em linha reta, dos vetores suporte ao hiperplano separador. A margem da formulação IMA$_1$, que minimiza a norma L_∞, é o somatório, em módulo, do valor das componentes, por isso sempre tem o maior dos valores na comparação. Já a

Tabela 5.2: Valores das normas e margens dos hiperplanos da Figura 5.1.

Pts.	$\|\|w\|\|_q$			γ		
	IMA_1	IMA_2	IMA_∞	IMA_1	IMA_2	IMA_∞
2	483,6	21,5	56,5	1,490	1,118	0,991
4	398,7	48,6	170,6	0,968	0,689	0,499
6	8,2	283,8	180,6	0,889	0,689	0,499
8	11,5	367,2	901,2	0,930	0,689	0,494
10	129,8	95,8	49,1	0,567	0,492	0,360
12	40,7	21,1	102,0	0,262	0,226	0,166

margem da formulação IMA_∞, que minimiza a norma L_1, é o valor, em módulo, da maior componente e, com isso, tem o menor dos valores.

Foi medida, também, a esparsidade do vetor w. Essa medição foi feita em função do número de componentes com valor superior ao valor do percentual aplicado ao valor da maior componente. Para os testes, os valores escolhidos foram 0,1%, 1% e 10%. A Tabela 5.3 mostra os resultados dos testes realizados com essas duas bases de dados.

Tabela 5.3: Comparação de valores para os algoritmos IMA_p e SMO.

Base	Algoritmo	Esparsidade			γ
		0,1%	1%	10%	
Mushroom	IMA_1	95	90	59	1,464
	IMA_2	88	78	52	0,365
	IMA_∞	13	9	9	0,099
	SMO		—		0,367
Sonar	IMA_1	60	60	56	0,017
	IMA_2	60	57	48	0,004
	IMA_∞	52	48	33	0,001
	SMO		—		0,004

Para problemas com uma esparsidade grande do vetor w, exemplo do problema Mushroom, o algoritmo IMA_∞ encontra facilmente componentes que não são importantes, ou seja, que são consideradas irrelevantes, podendo ser eliminadas do processo de classificação. Para problemas menos esparsos, caso do problema Sonar, essas eliminações de atributos se tornam um pouco mais difíceis.

Para se ter um mapeamento completo de uma base de dados, foi utilizada a base Iris com todos os valores dos pesos do vetor w (normalizados no intervalo $[0, 1]$) e das margens das normas L_1 e L_2 de cada subconjunto possível. A Figura 5.2 mostra um mapeamento completo dos subconjuntos possíveis dessa base de dados, onde os valores da esquerda são as margens na minimização da norma L_1 e os da direita são da norma L_2, e a Tabela 5.4 mostra os valores do vetor de pesos w para as normas L_1 e L_2. Círculos brancos indicam que o atributo não foi selecionado; círculos pretos

indicam sua seleção.

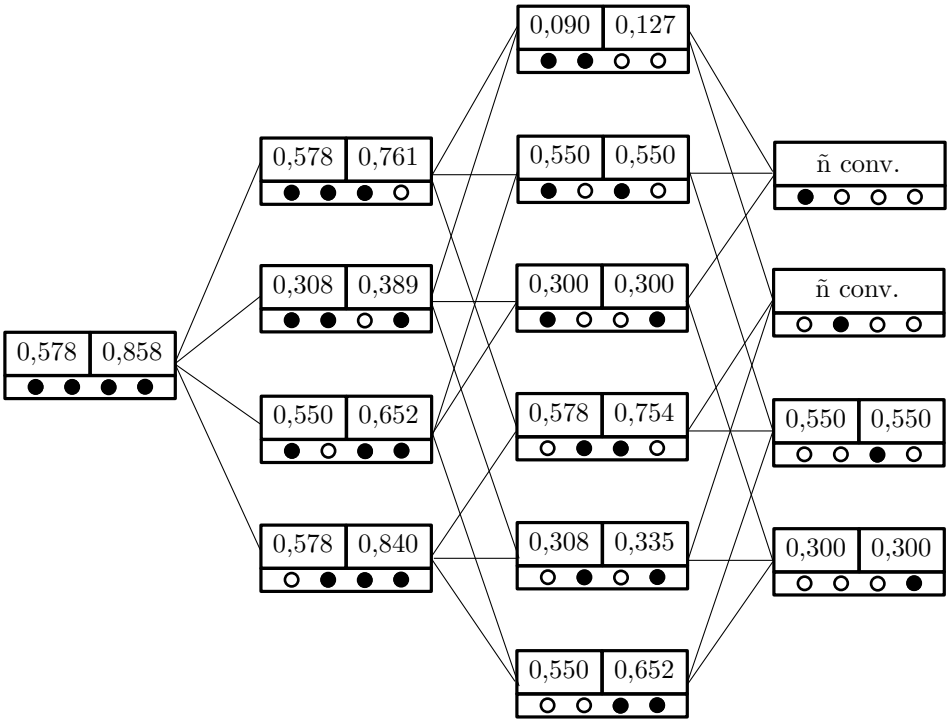


Figura 5.2: Mapeamento dos subconjuntos da base Iris.

Tabela 5.4: Comparação dos valores do vetor w para a base Iris.

Subconjunto	L_1					L_2				
	$w\,[1]$	$w\,[2]$	$w\,[3]$	$w\,[4]$	γ	$w\,[1]$	$w\,[2]$	$w\,[3]$	$w\,[4]$	γ
● ○ ○ ○	ñ	—	—	—	ñ	ñ	—	—	—	ñ
○ ● ○ ○	—	ñ	—	—	ñ	—	ñ	—	—	ñ
○ ○ ● ○	—	—	1,000	—	0,550	—	—	1,000	—	0,550
○ ○ ○ ●	—	—	—	1,000	0,300	—	—	—	1,000	0,300
● ● ○ ○	0,550	0,450	—	—	0,090	0,774	0,633	—	—	0,127
● ○ ● ○	0,002	—	0,998	—	0,550	0,037	—	0,999	—	0,550
● ○ ○ ●	0,001	—	—	0,999	0,300	0,001	—	—	0,999	0,300
○ ● ● ○	—	0,285	0,715	—	0,578	—	0,371	0,929	—	0,754
○ ● ○ ●	—	0,083	—	0,917	0,308	—	0,098	—	0,995	0,335
○ ○ ● ●	—	—	0,999	0,001	0,550	—	—	0,832	0,555	0,652
● ● ● ○	0,001	0,208	0,719	—	0,578	0,078	0,397	0,915	—	0,761
● ● ○ ●	0,001	0,084	—	0,915	0,308	0,341	0,361	—	0,868	0,389
● ○ ● ●	0,002	—	0,997	0,001	0,550	0,015	—	0,842	0,539	0,652
○ ● ● ●	—	0,285	0,714	0,001	0,578	—	0,360	0,826	0,433	0,840
● ● ● ●	0,001	0,007	0,991	0,001	0,578	0,135	0,401	0,796	0,434	0,858

É possível perceber que para as soluções com a norma L_1, o valor do atributo 3, que mais separa as bases, é bastante mais significativo em relação aos outros do que nas soluções com a norma L_2. Isso evidencia a maior esparsidade do vetor.

5.2.2 Base Artificial

Para criar um cenário frequentemente encontrado em bases de *microarrays*, foi gerada um distribuição normal de 20 dimensões seguindo os seguintes parâmetros:

$$\Sigma = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 5 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 5 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 5 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\ 5 & 5 & 5 & 0 & \cdots & 0 & 76 \end{pmatrix} \mu_+ = \begin{pmatrix} 1 \\ 1 \\ 1 \\ N(0, 1/2) \\ \vdots \\ N(0, 1/2) \\ 0 \end{pmatrix} \mu_- = \begin{pmatrix} -1 \\ -1 \\ -1 \\ N(0, 1/2) \\ \vdots \\ N(0, 1/2) \\ 0 \end{pmatrix},$$

onde Σ é a matriz de covariância, μ_+ e μ_- são os vetores médios para cada classe, e $N(0, 1/2)$ indica um valor aleatório proveniente de uma distribuição normal com média 0 e desvio padrão $1/2$. É importante notar que as características com valores médios ajustados em $N(0, 1/2)$ não são correlacionadas com quaisquer outras características e foram introduzidas para simular ruídos.

Com esta distribuição, tentou-se simular a correlação existente entre genes e ruídos obtidos a partir de genes sem informação. É possível observar que as características $1, 2, 3$ e 20 são correlacionadas, com o atributo 20 sendo bastante informativo quando combinado com qualquer um dos outros três. Desta forma, é interessante que um método de seleção de características encontre essa correlação.

A Figura 5.3 mostra amostras retiradas desta distribuição normal. As classes são representadas pelos símbolos $\times$ e $+$. Nesse conjunto, foram omitidas as características de ruídos e uma das primeiras três características correlacionadas. Como se pode perceber, os conjuntos $\{1, 2\}, \{1, 3\}, \{2, 3\}$ por si só não são bons conjuntos discriminatórios quando comparados com os três juntos.

A fim de testar o algoritmo AOS, e compará-lo com os algoritmos RFE e Golub, gerou-se 20 conjuntos de treinamento a partir desta distribuição, cada um composto de 20 variáveis e 1600 pontos (800 de cada classe). Além disso, 20 conjuntos de teste também foram gerados, cada um com 100 pontos de cada classe. Cada algoritmo foi usado para selecionar quatro características de cada conjunto de treinamento e, após a seleção, um conjunto de teste foi utilizado para testar o poder de classificação de um SVM nesse conjunto. Como os métodos em filtro não utilizam informações dos demais atributos para avaliar uma característica, o Golub não conseguiu por muitas vezes detectar a correlação entre as variáveis e não as selecionou diversas vezes. O RFE também não selecionou sempre os quatro atributos desejáveis. Em algumas situações, onde existia algum ruído um pouco maior, ele removeu a característica 20 antes de remover todos os ruídos e, como é um algoritmo míope, não conseguiu voltar atrás. Já o AOS foi capaz de encontrar o conjunto $\{1, 2, 3, 20\}$ em todas as

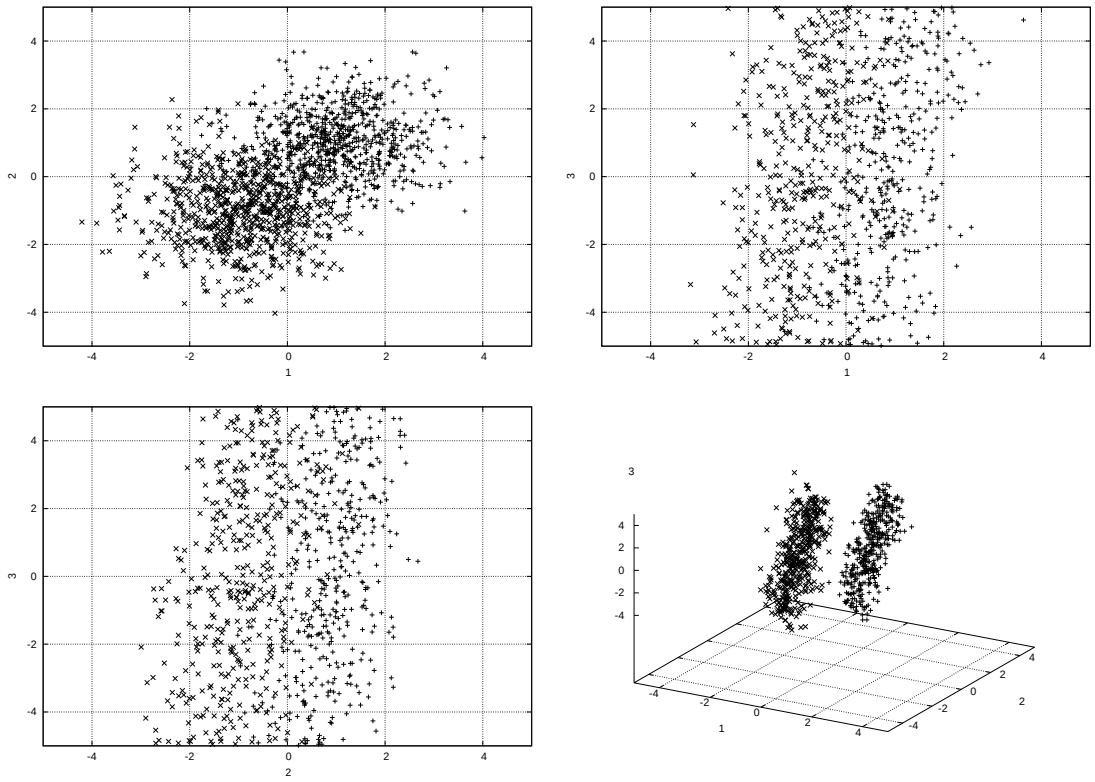


Figura 5.3: Correlação entre as variáveis da base artificial.

execuções, o que reforça o alto desempenho do algoritmo. A Tabela 5.5 mostra as margens e erros médios, das 20 execuções, para os três métodos.

Tabela 5.5: Comparação entre os métodos para a base artificial.

Método	γ	Erro
SVM	1,925	0,05%
Golub	0,781	8,92%
RFE	1,056	6,79%
AOS	1,604	0,00%

5.2.3 *K-fold Cross Validation*

A idéia principal do método *k-fold cross validation* (validação cruzada) é dividir o conjunto de treinamento M de tamanho n em k subconjuntos. Cada subconjunto M_i, para $1 \leq i \leq k$, será do mesmo tamanho (n/k). Antes da divisão do conjunto, todas as instâncias são embaralhadas. Então são gerados k modelos. O $i-$ésimo modelo é construído com o conjunto de treinamento $M - M_i$, que é a diferença entre o subconjunto de entrada e o subconjunto M_i. O modelo é então testado com o conjunto M_i. Isso se repete para cada i, $1 \leq i \leq k$. Por fim é calculada a média do erro para os k modelos, determinada pelos resultados obtidos com os testes.

Com essa metodologia, não é necessária a separação das instâncias em conjuntos de treinamento e testes antecipadamente. Todos os dados utilizados na geração são utilizados em ambos os papéis: treinamento e testes. A idéia do método foi proposta por MOSTELLER e TUKEY [76] e revisada por GEISSER [77] e WAHBA e WOLD [78]. Uma revisão dos métodos de validação cruzada é apresentada por BROWNE [79].

Em uma derivação desse método, [80], é utilizada a estratificação visando obter melhores resultados. Com a estratificação, é mantida a mesma proporcionalidade entre as classes em cada um dos k conjuntos, isso é, antes de dividir o conjunto de treinamento em k partes de tamanho n/k é verificada a porcentagem de instâncias de cada classe no conjunto de instâncias disponíveis. Então, quando o conjunto de treinamento é dividido, a porcentagem de instâncias de cada classe é mantida, e cada subconjunto conterá a mesma proporcionalidade entre as classes. KOHAVI [80] e MCLACHLAN *et al.* [81] sugerem um valor para k igual a 10.

Nos testes realizados, os conjuntos de dados foram divididos em $^2/_3$ para o processo de seleção de características e, consequentemente, para os treinamentos, e $^1/_3$ para o processo de teste (validação dos resultados). Para a validação dos conjuntos de treinamento, foi utilizado o valor do erro médio de 10 execuções de um 10-*fold cross validation*. Para comparações mais precisas, foram selecionados, para uma mesma base, sempre os mesmos conjuntos de treinamento e teste e sempre os mesmos 10 conjuntos para as validações cruzadas, preservando as sementes geradoras de aleatoriedade.

5.2.4 *Kernel*

Para a escolha do *kernel* que seria utilizado para as bases não linearmente separáveis, foram realizadas variações nos *kernels* polinomial e gaussiano (descritos a seguir), e nos seus parâmetros, e foram verificados os erros médios de 10 execuções de um 10-*fold* no conjunto de treinamento e o erro da validação no conjunto de teste. Para o *kernel* polinomial foram utilizados os graus, d, iguais a 2 e 3. Já para o gaussiano, foram utilizados valores de σ iguais a 0,01, 0,1, 1, 10 e 100. Os resultados foram obtidos através da execução do algoritmo AOS com as parametrizações "padrões", ou seja, com ramificação completa e sem nenhum tipo de poda. A tabela 5.6 mostra esses resultados. As definições dos *kernels* polinomial e gaussiano são, respectivamente:

$$K(x_i, x_j) = (\langle x_i \cdot x_j \rangle + 1)^d \tag{5.1}$$

$$K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \tag{5.2}$$

Tabela 5.6: Escolha do *kernel*.

Base	*Kernel*	Parâmetro	Atributos	10-*fold*	Valid.
Ionosphere	Polinomial	2	6	8,55%	10,26%
		3	5	9,38%	12,82%
	Gaussiano	0,01	3	22,26%	17,95%
		0,1	3	19,67%	15,38%
		1	3	10,74%	10,26%
		10	5	8,12%	12,82%
		100	7	11,54%	9,40%
WDBC	Polinomial	2	ñ conv.	—	—
		3	3	48,31%	36,32%
	Gaussiano	0,01	2	37,20%	37,37%
		0,1	2	36,68%	36,32%
		1	2	37,20%	37,37%
		10	2	37,47%	37,37%
		100	2	32,99%	36,32%
Bupa	Polinomial	2	ñ conv.	—	—
		3	ñ conv.	—	—
	Gaussiano	0,01	3	40,87%	40,87%
		0,1	3	40,87%	40,00%
		1	3	41,30%	40,87%
		10	3	37,39%	42,61%
		100	3	36,09%	41,74%
Pima	Polinomial	2	ñ conv.	—	—
		3	ñ conv.	—	—
	Gaussiano	0,01	3	34,77%	35,16%
		0,1	3	34,77%	35,16%
		1	3	34,57%	36,33%
		10	3	34,97%	42,58%
		100	3	36,13%	28,13%
Wine	Polinomial	2	ñ conv.	—	—
		3	3	56,44%	57,63%
	Gaussiano	0,01	2	40,30%	38,98%
		0,1	2	38,45%	35,59%
		1	2	35,01%	22,03%
		10	2	29,75%	22,03%
		100	2	22,20%	18,64%

De posse dos resultados, o *kernel* escolhido para a realização dos experimentos foi o gaussiano e o valor do parâmetro σ igual a 1.

5.2.5 Fator de Ramificação

Para a escolha dos parâmetros para o fator de ramificação e para as podas de profundidade e de corte, foram escolhidas seis bases e foi utilizado o RFE até a dimensão

20, para tornar a busca em todo o espaço possível. Com isso, tem-se 2^{20} possibilidades de solução para cada base. Excluindo-se o estado inicial, que não precisa entrar na fila, e o estado com nenhuma característica, tem-se 1.048.574 estados possíveis para bases com 20 atributos. Foram selecionadas duas bases sintéticas (Synthetic e Robot LP4), duas de *microarray* (Prostate e Colon) e duas não lineares (Ionosphere e Sonar – que é uma base linear com todas as características, mas não com somente 20). Para efeito de comparação, todos os valores de margens estão na norma L_2, ou seja, pelo valor da distância euclidiana.

Para a escolha de um fator de ramificação que gerasse boas soluções e fosse computacionalmente possível, foram realizados testes com os valores 2, 3, 5 e 20 (busca completa), além da execução do RFE. Foram comparadas as quantidades de candidatos inseridos (Ins.), reinseridos (Reins.) (estados com margem projetada que foram treinados e não tinham, de fato, uma maior margem) e expandidos (Exp.), o tamanho máximo da fila de estados (Max.) e a quantidade de nós que não precisou de treinamento ($\tilde{\text{N}}$ Trein.), além da dimensão final (Dim.), com seu valor de margem (γ). Foram realizados testes nas versões IMA_∞, que minimiza a norma L_1 do vetor, e IMA_2, que minimiza a própria norma L_2. As tabelas 5.7, 5.8 e 5.9 mostram essas variações no fator de ramificação.

Tabela 5.7: Fator de ramificação para as bases sintéticas.

Base	L_q	b	Ins.	Reins.	Exp.	Max.	$\tilde{\text{N}}$ Trein.	Dim.	γ
Synthetic	L_1	RFE	—	—	—	—	—	6	0,136
		2	1191	753	866	271	103	6	0,563
		3	10134	6884	6992	2599	1081	6	0,571
		5	127976	91577	70693	44962	19126	6	0,571
		20	690518	418068	263024	299125	256779	6	0,571
	L_2	RFE	—	—	—	—	—	7	0,819
		2	1342	995	976	297	142	6	0,514
		3	15031	11044	9875	3850	1676	6	0,571
		5	141063	97538	76825	42648	30621	6	0,571
		20	690922	429210	270130	279325	257890	6	0,571
LP4 Robot	L_1	RFE	—	—	—	—	—	7	0,458
		2	253	186	43	189	27	5	0,585
		3	3182	2187	601	2069	826	4	0,205
		5	36199	22671	7973	20453	12589	4	0,205
		20	864961	446326	348586	407498	393180	4	0,205
	L_2	RFE	—	—	—	—	—	7	0,342
		2	275	182	50	206	25	5	0,583
		3	3775	2582	750	2589	754	4	0,205
		5	45466	29706	11040	27765	12532	4	0,205
		20	877821	470892	313104	426118	406856	4	0,205

É possível visualizar que a norma L_1, para as bases linearmente separáveis, como

Tabela 5.8: Fator de ramificação para as bases de *microarrays*.

Base	L_q	b	Ins.	Reins.	Exp.	Max.	$\tilde{\text{N}}$ Trein.	Dim.	γ
Leukemia	L_1	RFE	—	—	—	—	—	4	23,306
		2	263	138	204	47	47	4	23,306
		3	2337	1375	1556	520	626	4	23,306
		5	41618	25128	23831	10582	14336	4	23,306
		20	928709	568739	485215	349554	354374	4	23,306
	L_2	RFE	—	—	—	—	—	6	72,498
		2	673	524	536	124	116	4	23,306
		3	7335	5221	5123	1388	1907	4	23,306
		5	80067	53283	48081	17665	25782	4	23,306
		20	938122	575796	505402	320345	362307	4	23,306
Prostate	L_1	RFE	—	—	—	—	—	7	62,346
		2	250	170	192	50	18	5	9,043
		3	2653	1698	1608	596	630	5	16,683
		5	50670	30528	25858	13324	17080	5	16,683
		20	756809	457765	323150	304999	293076	5	16,683
	L_2	RFE	—	—	—	—	—	7	23,108
		2	409	338	306	73	46	6	26,736
		3	5045	3679	3164	1077	1197	5	16,683
		5	67281	43557	34668	16238	22956	5	16,683
		20	774370	497827	345780	285740	306542	5	16,683

Tabela 5.9: Fator de ramificação para as bases não lineares.

Base	b	Ins.	Reins.	Exp.	Max.	$\tilde{\text{N}}$ Trein.	Dim.	γ
Sonar	RFE	—	—	—	—	—	4	0,002
	2	953	796	753	196	145	4	0,003
	3	9862	8221	7684	2201	1601	4	0,003
	5	116011	105616	99616	26892	10163	4	0,003
	20	1045569	1033415	1023637	316622	12154	4	0,003
Ionosphere	RFE	—	—	—	—	—	4	0,006
	2	717	668	614	102	39	4	0,009
	3	7623	6936	5700	1420	641	4	0,012
	5	98717	91631	75947	23413	6828	4	0,012
	20	1043881	1014712	986521	343921	28715	4	0,012

mencionado no capítulo 3 e mostrado na seção 5.2.1, é de fato melhor para seleção de características. Para ramificações pequenas, ou para o RFE, ela alcança soluções melhores, ou pelo menos iguais. Já para ramificações grandes, aonde a busca é mais ampla, ela encontra as mesmas soluções da norma L_2, porém de forma mais rápida. É possível visualizar, também, que não é necessário mais do que a geração de 3 hipóteses por cada estado escolhido, para se ter as mesmas soluções da busca exaustiva. Não é possível garantir que, para bases com mais de 20 dimensões, o valor 3 será suficiente para encontrar os melhores resultados, mas é possível prever que

sim, uma vez que existe grande possibilidade de, para bases maiores, serem inseridos mais do que um estado com dimensão 20 na fila de abertos, aumentando, assim, o espaço de busca. Com isso, foram escolhidos a norma L_1 para as bases lineares, e o valor 3 para o fator de ramificação b para todas as bases.

5.2.6 Podas

Para as escolhas das podas, para tornar o processo mais rápido, foram realizados testes com os valores 2, 3 e 5 para cada tipo de poda, além de combinações dos valores 2 e 3 para os dois tipos juntos. Foram comparadas as quantidades de candidatos inseridos, reinseridos e expandidos, o tamanho máximo da fila, a quantidade de nós não treinados e a quantidade de nós que foram podados da fila (Podas), além da dimensão final, com seu valor de margem. As tabelas 5.10, 5.11 e 5.12 mostram as variações das podas de profundidade e de corte.

Tabela 5.10: Podas para as bases sintéticas.

Base	p	c	Ins.	Reins.	Exp.	Max.	Ñ Trein.	Podas	Dim.	γ
Synthetic	—	—	10134	6884	6992	2599	1081	0	6	0,571
	2	—	483	308	267	149	106	16	6	0,571
	3	—	1990	1383	1291	524	274	30	6	0,571
	5	—	2996	2054	1944	780	419	58	6	0,571
	—	2	214	139	92	66	58	33	6	0,571
	—	3	502	348	266	165	108	16	6	0,571
	—	5	679	457	373	220	144	10	6	0,571
	2	2	217	125	103	70	67	16	6	0,571
	2	3	437	290	235	127	104	20	6	0,571
	3	2	175	91	81	52	59	14	6	0,571
	3	3	370	232	198	101	95	23	6	0,571
Robot LP4	—	—	3182	2187	2069	601	826	0	4	0,205
	2	—	1833	1187	1172	351	425	18	4	0,205
	3	—	2123	1418	1374	409	442	74	4	0,205
	5	—	63	34	22	13	13	27	5	0,583
	—	2	191	116	88	37	61	36	4	0,205
	—	3	320	190	162	59	109	40	4	0,205
	—	5	590	356	345	105	185	21	4	0,205
	2	2	179	96	81	36	59	35	4	0,205
	2	3	299	166	158	60	101	31	4	0,205
	3	2	186	102	85	37	58	38	4	0,205
	3	3	257	145	134	51	82	27	4	0,205

Como foi utilizado um fator de ramificação de valor 3, uma poda de profundidade alta pode causar uma eliminação excessiva ou inserções desnecessárias. Isso porque fazer uma grande descida míope pode abrir um caminho que não seria escolhido

Tabela 5.11: Podas para as bases de *microarrays*.

Base	p	c	Ins.	Reins.	Exp.	Max.	$\tilde{\text{N}}$ Trein.	Podas	Dim.	γ
Colon	—	—	2337	1375	1556	520	626	0	4	23,306
	2	—	2456	1423	1651	543	691	40	4	23,306
	3	—	1810	1083	1171	363	458	94	4	23,306
	5	—	1771	1051	1152	346	472	78	4	23,306
	—	2	269	165	133	49	71	63	4	23,306
	—	3	654	368	363	140	228	43	4	23,306
	—	5	2233	1315	1446	500	618	9	4	23,306
	2	2	298	159	141	63	92	63	4	23,306
	2	3	511	269	286	103	177	43	4	23,306
	3	2	164	72	74	35	57	39	4	23,306
	3	3	532	296	298	122	177	35	4	23,306
Prostate	—	—	2653	1698	1608	596	630	0	5	16,683
	2	—	2382	1257	1413	417	851	19	5	16,683
	3	—	2307	1434	1410	505	559	1	5	16,683
	5	—	2827	1790	1732	672	652	0	5	16,683
	—	2	211	144	98	53	49	43	6	32,330
	—	3	666	359	343	126	251	59	5	16,683
	—	5	2075	1127	1242	383	726	17	5	16,683
	2	2	215	135	100	54	45	46	5	16,683
	2	3	845	454	444	161	301	87	5	16,683
	3	2	139	81	59	37	28	35	5	16,683
	3	3	540	352	284	135	116	40	5	16,683

pelos candidatos "normais" ou então a atualização do valor limite pode eliminar candidatos que gerariam boas soluções. Da mesma forma, escolher uma baixa poda de corte, deixando somente candidatos de poucas dimensões, pode ocasionar cortes demais e fazer com que alguns candidatos "bons" sejam eliminados da fila.

Os dois parâmetros combinados geraram sempre os mesmos resultados das execuções sem podas. A combinação que obteve as menores quantidades de estados gerados foi a com valor 2 para ambos os parâmetros. Por esse motivo, foi escolhido uma poda de profundidade p igual a 2, assim como uma poda de corte c também igual a 2.

5.2.7 Critérios de Ramificação e Medidas de Avaliação

Para as escolhas do critério de ramificação e da medida de avaliação dos candidatos, foram realizados testes com as opções fornecidas no Capítulo 4, ou seja, foram utilizadas as medidas w (1), $w \cdot R$ (2), w/C (3), $w \cdot R/C$ (4), e $w \cdot G$ (5) para as variações do critério de ramificação (r) e as medidas γ e $\gamma \cdot C$ para as variações da medida, ou função, de avaliação (f). Foram realizadas as 10 combinações possíveis dessas medidas e testadas em quatro bases de dados, sendo três lineares (Robot

Tabela 5.12: Podas para as bases não lineares.

Base	p	c	Ins.	Reins.	Exp.	Max.	Ñ Trein.	Podas	Dim.	γ
Sonar	—	—	9862	8221	7684	2201	1601	0	4	0,003
	2	—	11125	9232	8684	2457	1816	0	4	0,003
	3	—	14367	12042	8619	3166	2228	0	4	0,003
	5	—	17453	14836	11169	3997	2493	0	4	0,003
	—	2	6650	5487	13627	1545	1141	256	4	0,003
	—	3	9412	8208	4756	2118	1541	8	4	0,003
	—	5	9862	8221	7251	2201	1601	0	4	0,003
	2	2	7311	6094	7684	1584	1153	312	4	0,003
	2	3	10886	9239	5292	2336	1575	93	4	0,003
	3	2	7379	6126	8482	1662	1183	316	4	0,003
	3	3	11815	9994	5380	2519	1732	105	4	0,003
Ionosphere	—	—	7623	6936	5700	1420	641	0	4	0,012
	2	—	9548	8765	7273	1707	720	12	4	0,012
	3	—	11435	10436	8591	2109	941	55	4	0,012
	5	—	11694	10649	8786	2164	972	48	4	0,012
	—	2	1075	934	578	301	140	86	4	0,012
	—	3	2760	2410	1711	699	346	92	4	0,012
	—	5	6418	5831	4635	1313	561	26	4	0,012
	2	2	1352	1130	791	353	194	60	4	0,012
	2	3	2114	1804	1330	540	284	29	4	0,012
	3	2	1370	1140	800	352	192	65	4	0,012
	3	3	2154	1837	1389	530	281	29	4	0,012

LP4, Mushroom e Prostate) e uma não linear (Ionosphere). Foram comparadas as quantidades de candidatos inseridos, reinseridos, expandidos, não treinados, não inseridos (Ñ Ins.) e podados, além da dimensão final, com seu valor de margem, e o erro médio de 10 execuções de um 10-*fold*. As Tabelas 5.13 e 5.14 mostram as variações do fator de ramificação.

É possível constatar que a utilização da função que utiliza a distância dos centros só é válida quando combinada a critérios de ramificação que também utilizam essa medida. Isso se deve ao fato de que, para os demais critérios, as hipóteses geradas não têm um comprometimento com ambas as medidas, podendo ter, assim, um dos valores muito baixo e não sendo nem inseridos na fila. Porém, mesmo essa função sendo interessante associada com alguns critérios, ela gerou hipóteses demais e, ainda assim, não foi superior do que a medida que utiliza somente o valor da margem como seleção. Isso porque, para uma solução SVM, é mais importante uma maior distância entre amostras de classes opostas mais próximas do que entre as médias das classes. Talvez a utilização de uma medida que pesasse mais a margem, mas que desse uma importância à distância dos centros, fosse interessante. O critério que associa o *score* Golub, como era de esperar, gerou um resultado satisfatório para algumas bases e

Tabela 5.13: Critérios de ramificação e medidas de avaliação para as bases lineares.

Base	f	r	Ins.	Reins.	Exp.	$\tilde{\text{N}}$ Trein.	$\tilde{\text{N}}$ Ins.	Podas	Dim.	γ	10-*fold*
Robot LP4	γ	1	558	259	203	141	0	272	4	0,246	3,77%
		2	517	244	181	99	0	275	4	0,246	3,77%
		3	407	160	141	89	0	240	4	0,331	3,48%
		4	518	226	187	130	0	250	4	0,246	3,77%
		5	389	144	137	100	19	207	4	0,331	3,48%
	$\gamma \cdot C$	1	106	0	74	6	148	31	5	1,057	4,25%
		2	98	0	67	8	134	30	5	1,057	4,25%
		3	651	262	270	165	13	307	4	0,331	3,48%
		4	658	256	269	188	1	307	4	0,331	3,48%
		5	145	0	76	50	152	69	5	0,087	6,96%
Mushroom	γ	1	267	55	73	66	4	190	5	0,224	0,00%
		2	288	73	84	62	12	204	5	0,224	0,00%
		3	258	63	72	58	5	182	5	0,224	0,00%
		4	264	64	74	65	5	186	5	0,224	0,00%
		5	262	56	68	86	10	194	5	0,224	0,00%
	$\gamma \cdot C$	1	133	0	96	19	192	36	5	0,224	0,00%
		2	140	0	101	19	202	39	5	0,224	0,00%
		3	2072	565	1351	555	57	709	5	0,224	0,00%
		4	1287	274	743	412	39	524	5	0,224	0,00%
		5	177	0	96	65	192	81	5	0,224	0,00%
Prostate	γ	1	858	522	380	143	0	389	4	18,912	2,10%
		2	959	623	382	142	0	484	4	18,912	2,10%
		3	903	604	383	103	0	467	4	18,912	2,10%
		4	821	476	327	162	0	397	4	18,912	2,10%
		5	688	398	263	117	0	375	4	14,414	2,95%
	$\gamma \cdot C$	1	120	0	89	11	178	31	7	25,262	9,10%
		2	124	0	92	11	184	31	7	9,697	10,48%
		3	937	256	356	376	14	526	5	3,624	5,40%
		4	982	257	366	390	6	576	5	3,624	5,40%
		5	166	0	89	58	178	77	7	25,262	9,10%

para outras não, sendo, por isso, descartado. Já dentro dos critérios associados à função que utiliza os valores de margem, o critério que maximiza a distância dos centros, juntamente com a maximização da margem (retirada da menor componente do vetor w), foi o que gerou as soluções com os menores erros associados. Por isso, esse critério foi escolhido.

5.2.8 Resultados Finais

Foram comparadas as soluções dos métodos Golub, *Nearest Shrunken Centroid*, *Recursive Feature Elimination* e *Admissible Ordered Search*, além da solução sem a eliminação de variáveis (linha SVM).

Tabela 5.14: Critérios de ramificação e medidas de avaliação para a base Ionosphere.

Base	f	r	Ins.	Reins.	Exp.	Ñ Trein.	Ñ Ins.	Podas	Dim.	γ	10-*fold*
Ionosphere	γ	1	2081	1498	1296	536	0	148	3	0,001	15,87%
		2	2224	1614	1412	563	0	139	3	0,001	15,87%
		3	593	462	330	69	0	67	3	0,002	10,74%
		4	834	535	522	237	0	74	3	0,001	14,92%
		5	1142	784	661	292	0	157	3	0,001	13,24%
	$\gamma \cdot C$	1	74	0	49	19	98	24	4	0,001	16,60%
		2	73	0	49	18	98	24	4	0,001	16,60%
		3	8085	6011	5062	2010	0	979	3	0,001	15,87%
		4	9335	7190	6130	2085	162	1052	3	0,001	15,87%
		5	87	0	48	32	96	39	4	0,001	16,60%

Para o Golub, foram removidas as características segundo seus *scores* até não se ter mais uma separabilidade linear ou, para as bases não lineares, até o algoritmo de classificação não mais convergir.

Para o NSC, foi utilizada a versão do pacote 'pamr', para a linguagem R, referenciada por TIBSHIRANI *et al.* [82]. Utilizou-se, conforme sugerem os autores, um valor para o parâmetro Δ (*threshold*) que gerasse a menor quantidade de atributos, sem ocorrer erros demasiados.

Para os testes feitos nos algoritmos RFE e AOS foi utilizado, como mostrado nas seções anteriores, o Algoritmo de Margem Incremental, na sua versão IMA_∞ para as bases lineares e o IMA dual para as bases não lineares.

Para o RFE, o processo utilizado foi a remoção de uma característica por vez até o problema ficar não linearmente separável, ou o algoritmo de classificação não ter convergência, para as bases não linearmente separáveis.

Com base nos resultados apresentados nas seções anteriores, os parâmetros escolhidos para as execuções do AOS foram:

- Fator de ramificação b: 3;
- Profundidade da poda p: 2;
- Profundidade do corte c: 2;
- Critério de ramificação: w/c;
- Medida de avaliação: γ.

Para as bases de *microarrays*, foi utilizado o RFE até a dimensão 100, para tornar a busca gerenciável. A partir desta quantidade de atributos, então, foi utilizado o AOS, com as mesmas parametrizações definidas acima. Foi realizado um teste com a base Colon, que possui a menor quantidade de atributos dentre as bases

de *microarrays*, com o conjunto total de atributos e se obteve o mesmo resultado que o processo utilizando o RFE até a dimensão 100 e só depois utilizando o AOS. Esse resultado, que não garante que ocorrerá para todas as demais bases, mostra que a busca se torna mais difícil quando o problema possui menos atributos, sendo necessária, nesse ponto, uma busca mais ampla.

A fila de estados foi implementada utilizando uma estrutura de *heap*, uma vez que, a cada escolha, somente é necessário o candidato com a maior margem e não que a lista esteja totalmente ordenada. Com isso, o desempenho do algoritmo é aumentado. A tabela de dispersão (*hash*) foi implementada utilizando a seguinte função de espalhamento (dispersão):

$$h\left(k\right) = \left(\sum_{i} f_i^2\right) \bmod n, \tag{5.3}$$

onde f_i são as características pertencentes a um determinado conjunto e n é um número primo, para evitar posições repetidas. Nos testes foi utilizado um valor de n igual a 161.387. Para cada uma das k posições, foi implementado um vetor com 100 posições para evitar colisões.

Os algoritmos foram implementados na linguagem C, utilizando o compilador GCC, com exceção do NSC como já mencionado, e testados em um computador Intel Mobile Core i7 com 6GB de memória RAM.

Os resultados encontrados para os métodos são apresentados na Tabela 5.15.

Os resultados mostram que o algoritmos AOS foi melhor, ou pelo menos igual, em todas as bases testadas. Sempre conseguiu uma menor quantidade, ou a mesma, de atributos que os demais métodos, com exceção em relação ao NSC em 3 bases, Mushroom (que o NSC obteve um subconjunto menor, porém com uma taxa de erro alta), Prostate e Sonar. Gerou, ainda, na maioria das bases, menores erros de validação cruzada e de validação no conjunto de teste.

Para a base WDBC, em que o NSC gerou taxas de erros inferiores, foi realizada uma verificação com um classificador SVM, para a comparação ficar mais real. No subconjunto em questão foi detectado um erro de 37,20% de 10-*fold* e 37,37% de teste, ou seja, os mesmos erros do AOS, sendo que o AOS chegou em um subconjunto com somente 2 atributos, contra 5 do NSC, mostrando que, como método de seleção, o AOS foi superior.

Para a base sintética Robot LP4, o NSC classificou todas as amostras como sendo de uma mesma classe. Com isso, ele obteve exatamente a quantidade de amostras da outra classe como erros de 10-$fold$ e de validação.

É importante observar que para a base Wine, tanto o NSC quanto o AOS chegaram no mesmo subconjunto de atributos, porém pelo NSC esse subconjunto obteve

Tabela 5.15: Comparação entre os métodos.

Método	Base	F	γ	10-*fold*	Valid.	Base	F	γ	10-*fold*	Valid.
SVM	Mushroom	98	0,366	0,00%	0,00%	Synthetic	60	7,721	0,03%	0,50%
Golub		39	0,236	0,00%	0,00%		35	0,984	5,10%	7,00%
NSC		3	—	9,70%	11,00%		6	—	15,25%	17,50%
RFE		5	0,224	0,00%	0,00%		7	0,618	0,93%	3,50%
AOS		5	0,224	0,00%	0,00%		5	0,131	0,35%	4,00%
SVM	Leukemia	7129	18181,530	0,00%	8,33%	LP4 Robot	90	7,028	17,84%	12,82%
Golub		5	97,609	8,60%	12,50%		17	0,062	22,23%	10,26%
NSC		4	—	6,25%	12,50%		5	—	20,51%	20,51%
RFE		4	1568,592	3,80%	12,50%		7	0,941	6,57%	7,69%
AOS		2	177,635	0,60%	25,00%		4	0,331	3,48%	2,56%
SVM	Colon	2000	1529,516	16,55%	19,05%	Sonar	60	0,007	27,21%	24,29%
Golub		9	59,396	19,30%	19,05%		41	0,001	28,35%	27,14%
NSC		7	—	21,95%	19,05%		9	—	31,16%	28,57%
RFE		5	116,788	8,40%	14,29%		25	0,003	13,59%	30,00%
AOS		4	41,502	7,45%	28,57%		21	0,001	12,20%	30,00%
SVM	Breast	12625	5222,205	26,50%	12,50%	Ionosphere	34	0,095	6,03%	12,82%
Golub		2	191,930	8,50%	12,50%		7	0,015	11,12%	10,26%
NSC		2	—	43,75%	50,00%		13	—	17,95%	22,22%
RFE		3	440,323	13,00%	25,00%		4	0,001	12,53%	13,68%
AOS		2	400,317	0,00%	12,50%		3	0,002	10,74%	10,26%
SVM	Prostate	12600	529,011	14,82%	5,88%	WDBC	30	0,053	37,20%	37,37%
Golub		15	16,143	17,43%	8,82%		5	0,046	38,13%	37,89%
NSC		3	—	13,24%	5,88%		5	—	11,87%	7,37%
RFE		6	24,496	8,00%	5,88%		2	0,052	37,20%	37,37%
AOS		4	18,912	2,10%	2,94%		2	0,052	37,20%	37,37%
SVM	DLBCL	5468	14688,808	2,93%	7,69%	Wine	13	0,093	40,30%	38,98%
Golub		11	479,156	14,43%	19,23%		3	0,093	35,52%	25,42%
NSC		10	—	15,68%	15,38%		2	—	12,61%	10,17%
RFE		2	156,619	3,73%	11,54%		3	0,093	40,30%	37,29%
AOS		2	269,738	0,17%	7,69%		2	0,088	35,01%	22,03%

erros menores. Isso se deve ao classificador do NSC e não pelo método de seleção, ou seja, para essa base de dados, os dois algoritmos obtiveram a mesma solução.

Para as bases em que o AOS obteve uma menor quantidade de características, porém um erro de 10-*fold* e/ou de validação superior, foram feitas comparações da solução do AOS com a dimensão alcançada pelo RFE. A Tabela 5.16 mostra essas comparações. É possível perceber que o AOS sempre encontra uma solução melhor, com margem superior e erros inferiores (ou iguais), com a mesma quantidade de atributos que a encontrada pelo RFE.

Para testar o poder do classificador dual nas bases lineares, utilizou-se o AOS, a partir dos resultados lineares finais, em algumas bases para tentar a retirada

Tabela 5.16: Comparação entre o AOS e o RFE para a mesma dimensão.

Base	F	RFE			AOS		
		γ	10-*fold*	Valid.	γ	10-*fold*	Valid.
Leukemia	4	1568,592	3,80%	12,50%	1836,574	3,55%	8,33%
Colon	5	116,788	8,40%	14,29%	218,454	2,25%	14,29%
Synthetic	7	0,618	0,93%	3,50%	1,744	0,00%	3,50%

de ainda mais características. Dentre essas bases estão as que o NSC conseguiu um subconjunto com menos atributos que o AOS, com exceção da Mushroom que, mesmo no espaço dual, o AOS não conseguiu retirar mais características. A Tabela 5.17 mostra os resultados para essas bases, comparando o conjunto total de atributos com os conjuntos selecionados pelos métodos linear (Lin.) e não linear (Ñ Lin.).

Tabela 5.17: Comparação entre o AOS linear e não linear.

Base	Método	SVM				AOS			
		F	γ	10-*fold*	Valid.	F	γ	10-*fold*	Valid.
Prostate	Lin.	12600	529,011	14,82%	5,88%	4	18,912	2,10%	2,94%
	Ñ Lin.		0,121	48,57%	50,00%	2	0,121	45,90%	50,00%
Colon	Lin.	2000	1529,516	16,55%	19,05%	4	41,502	7,45%	28,57%
	Ñ Lin.		0,165	34,00%	38,10%	2	0,165	34,00%	38,10%
Sonar	Lin.	60	0,007	27,21%	24,29%	21	0,001	12,20%	30,00%
	Ñ Lin.		0,090	13,95%	8,57%	4	0,001	17,97%	25,71%
Synthetic	Lin.	60	7,721	0,03%	0,50%	5	0,131	0,35%	4,00%
	Ñ Lin.		0,050	50,00%	50,00%	2	0,050	5,42%	6,00%
Robot LP4	Lin.	90	7,028	17,84%	12,82%	4	0,331	3,48%	2,56%
	Ñ Lin.		0,140	20,36%	20,51%	2	0,140	9,04%	7,69%

Com base nas informações da tabela, conclui-se que, para as bases lineares, não é tão interessante a remoção de mais características do que as que compõem um subconjunto linearmente separável. A utilização de remoções no espaço dual remove poucos atributos a mais e gera erros superiores aos dos subconjuntos encontrados com classificadores lineares. A exceção fica por conta da base Sonar, onde a remoção com um classificador não linear removeu uma quantidade significativa de atributos a mais e obteve erros similares aos gerados com um classificador linear.

É possível concluir que o algoritmo que foi proposto por esse trabalho gerou resultados bastante satisfatórios, tanto para classificadores lineares, em espacial o IMA com a formulação L_∞, quanto para classificadores não lineares.

O algoritmo permite uma busca muito mais ampla do que uma busca míope, que não possui a capacidade de realizar *backtracking*. Com essa busca, mesmo não sendo de fato completa, ele é capaz de encontrar subconjuntos de atributos altamente discriminatórios, removendo sempre uma quantidade bastante significativa de características.

Capítulo 6

Conclusões

I could feel a new kind of intelligence
across the table.

Garry Kasparov

O problema de seleção de características não possui uma solução trivial. Para cada tipo de problema, um determinado processo é mais adequado do que outro. E em cada problema se consegue eliminar mais ou menos características.

Nesse trabalho foi introduzido um novo algoritmo para a seleção de características, denominado *Admissible Ordered Search* (AOS). O algoritmo utiliza critérios e medidas de qualidade provenientes de classificadores de larga margem e explora com eficiência o espaço de possibilidades. Esse processo é um método bastante eficaz de seleção de características, pois o mesmo algoritmo gera os subconjuntos candidatos e executa o indutor sobre eles, selecionando sempre o melhor estado em todas as dimensões.

Como classificador de larga margem, o principal algoritmo utilizado foi o *Incremental Margin Algorithm* (IMA), na sua versão primal com suas formulações arbitrárias, destaque para a formulação L_∞, e na sua versão dual.

Como pode ser observado, o AOS apresentou resultados bastante satisfatórios. Ele foi superior aos demais métodos testados em todos os experimentos, sempre obtendo margens iguais ou superiores e erros de generalização iguais ou inferiores.

O algoritmo, além de permitir encontrar o subconjunto de maior margem em cada dimensão, também é capaz de encontrar, dentre esses subconjuntos, a solução que gera a menor quantidade de erros, ou seja, o subconjunto mais discriminatório.

É possível fazer um comparativo com o supercomputador da IBM, Deep Blue, que em 1997 ganhou do campeão mundial de xadrez, Garry Kasparov, feito que, à época, parecia ser impossível. O computador realmente não era capaz de prever o conjunto total de jogadas, mas com a busca heurística e as podas que possuía foi capaz de realizar sempre jogadas muito boas, conseguindo, assim, vitórias sobre seu

oponente, mostrando que mesmo para problemas com soluções ótimas "impossíveis" é bastante possível encontrar soluções muito interessantes.

O algoritmo AOS combina técnicas de duas grandes áreas da Inteligência Artificial (IA), a IA Cognitiva, com a busca de caminhos, e IA Conexionista, através do Aprendizado de Máquinas, nesse caso as Máquinas de Vetores Suporte, abrindo possibilidades de estudos nessa combinação.

6.1 Trabalhos Futuros

Como sugestão para o sequenciamento do trabalho, é possível realizar a tentativa da utilização do algoritmo AOS para o problema de seleção de características em regressão, utilizando a versão do IMA Regressor, [83].

Outra tentativa é a de se conseguir uma formulação arbitrária, principalmente a L_∞, no espaço de características para a sua utilização juntamente com a versão dual do algoritmo IMA. Sua solução se tornaria interessante para a remoção de atributos nas bases não linearmente separáveis.

É sabido que a escolha do melhor *kernel* (tipo e parâmetro) varia de base para base. Então uma sugestão é fazer uso de estudos como os realizados por WU e WANG [84], DEBNATH e TAKAHASHI [85] e SCHITTKOWSKI [86] para escolher o *kernel* mais apropriado para as bases não lineares.

Referências Bibliográficas

[1] ROSENBLATT, F. "The perceptron: a probabilistic model for information storage and organization in the brain", *Psychological Review*, v. 65, pp. 386–408, 1958.

[2] BLOCK, H. "The perceptron: a model for brain functioning", *Reviews of Modern Physics*, v. 34, pp. 123–135, 1962.

[3] NOVIKOFF, A. B. "On convergence proofs for perceptrons". In: *Proceedings of the Symposium on the Mathematical Theory of Automata*, v. 12, pp. 615–622, 1963.

[4] AGMON, S. "The relaxation method for linear inequalities", *Canadian Journal of Mathematics*, v. 6, n. 3, pp. 382–392, 1954.

[5] MOTZKIN, T. S., SCHOENBERG, I. J. "The relaxation method for linear inequalities", *Canadian Journal of Mathematics*, v. 6, n. 3, pp. 393–404, 1954.

[6] MINSKY, M., PAPERT, S. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA, USA, MIT Press, 1969.

[7] AIZERMAN, A., BRAVERMAN, E. M., ROZONER, L. I. "Theoretical foundations of the potential function method in pattern recognition learning", *Automation and Remote Control*, v. 25, pp. 821–837, 1964.

[8] BOSER, B. E., GUYON, I. M., VAPNIK, V. N. "A training algorithm for optimal margin classifiers". In: *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, pp. 144–152, New York, NY, USA, 1992. ACM Press.

[9] VAPNIK, V. N. *The nature of statistical learning theory*. New York, NY, USA, Springer-Verlag New York, Inc., 1995.

[10] TIKHONOV, A. N., ARSENIN, V. Y. *Solutions of Ill-posed problems*. Washington, D.C., W.H. Winston, 1977.

[11] WIDROW, B., HOFF, M. E. "Adaptive switching circuits", *Institute of Radio Engineers, Western Electronics Show and Convention*, v. Part 4, pp. 96–104, 1960.

[12] MERCER, J. "Functions of positive and negative type and their connection with the theory of integral equations", *Philosophical Transactions of the Royal Society, London*, v. 209, pp. 415–446, 1909.

[13] SCHÖLKOPF, B., HERBRICH, R., SMOLA, A. J. "A Generalized Representer Theorem". In: *In Proceedings of the Annual Conference on Computational Learning Theory*, pp. 416–426, 2001.

[14] DUDA, R. O., HART, P. E., STORK, D. G. *Pattern Classification*. 2. ed. New York, Wiley, 2001.

[15] LEITE, S. C., FONSECA NETO, R. "Incremental margin algorithm for large margin classifiers", *Neurocomputing*, v. 71, pp. 1550–1560, 2008.

[16] KIVINEN, J., SMOLA, A. J., WILLIAMSON, R. C. "Large Margin Classification for Moving Targets". In: *Algorithmic Learning Theory, 13th International Conference, ALT 2002, Lübeck, Germany*, v. 2533, *Lecture Notes in Artificial Intelligence*, pp. 113–127. Springer, 2002.

[17] KIVINEN, J., SMOLA, A. J., WILLIAMSON, R. C. "Online Learning with Kernels", *IEEE Transactions on Signal Processing*, v. 52, n. 8, 2004.

[18] HERBRICH, R. *Learning Kernel Classifiers: Theory and Algorithms*. Adaptive computation and machine learning. Cambridge, Massachusetts, MIT Press, 2002.

[19] SHAWE-TAYLOR, J., CRISTIANINI, N. "Margin Distribution and Soft Margin". In: *Advances in Large Margin Classifiers*, cap. 2, pp. 5–16, Cambridge, MA, MIT Press, 1999.

[20] SHAWE-TAYLOR, J., CRISTIANINI, N. *Kernel Methods for Pattern Analysis*. New York, NY, Cambridge University Press, 2004.

[21] LAMBERT, P. F. "Designing pattern categorizers with extremal paradigm information". In: Watanabe (Ed.), *Methodologies of Pattern Recognition*, pp. 359–391, New York, NY, 1969. Academic Press.

[22] VAPNIK, V. N., CHERVONENKIS, A. Y. *Theory of Pattern Recognition*. Moscow, Nauka, 1974.

[23] VAPNIK, V. *Estimation of Dependences Based on Empirical Data: Springer Series in Statistics*. Secaucus, NJ, USA, Springer-Verlag New York, Inc., 1982.

[24] LITTLESTONE, N., WARMUTH, M. *Relating data compression and learnability*. Relatório técnico, University of California Santa Cruz, 1986.

[25] WAHBA, G. "Soft and Hard Classification by Reproducing Kernel Hilbert Space Methods", *Proceedings of the National Academy of Sciences USA*, v. 99, n. 26, pp. 16524–16530, 2002.

[26] GIROSI, F. "An Equivalence Between Sparse Approximation and Support Vector Machines", *Neural computation*, v. 10, n. 6, pp. 1455–1480, 1998.

[27] EVGENIOU, T., PONTIL, M., POGGIO, T. "Regularization Networks and Support Vector Machines", *Advances in Computational Mathematics*, v. 13, n. 1, pp. 1–50, 2000.

[28] MANGASARIAN, O. L. "Linear and Nonlinear Separation of Patterns by Linear Programming", *Operations Research*, v. 13, pp. 444–452, 1965.

[29] BENNETT, K. P., MANGASARIAN, O. L. "Robust Linear Programming Discrimination of Two Linearly Inseparable Sets", *Optimization Methods and Software*, v. 1, n. 1, pp. 23–34, 1992.

[30] GENTILE, C. "A new approximate maximal margin classification algorithm", *Journal of Machine Learning Research*, v. 2, pp. 213–242, 2001.

[31] FREUND, Y., SCHAPIRE, R. "A short introduction to boosting", *Japonese Society for Artificial Intelligence*, v. 14, n. 5, pp. 771–780, 1999.

[32] GUYON, I., STORK, D. "Linear Discriminant and Support Vector Classifiers". In: Smola, A., Bartlett, P., Schölkopf, B., et al. (Eds.), *Advances in Large Margin Classifiers*, pp. 147–169, Cambridge, MA, 2000. MIT Press.

[33] CORTES, C., VAPNIK, V. "Support-vector networks", *Machine Learning*, v. 20, n. 3, pp. 273–297, 1995.

[34] SMOLA, A. J., SCHÖLKOPF, B. "On a Kernel-based Method for Pattern Recognition, Regression, Approximation and Operator Inversion", *Algorithmica*, v. 22, pp. 211–231, 1998.

[35] MANGASARIAN, O. L. "Multi-surface method of pattern separation", *IEEE Transactions on Information Theory*, v. IT-14, pp. 801–807, 1968.

[36] VILLELA, S. M., FONSECA NETO, R., LEITE, S. C., et al. "Classificador de Máxima Margem com Norma Arbitrária: Formulação, Algoritmo e Resultados". In: *IX CBRN - Congresso Brasileiro de Redes Neurais / Inteligência Computacional*, Ouro Preto, MG, 2009.

[37] MANGASARIAN, O. L. "Arbitrary-norm separating plane", *Operations Research Letters*, v. 24, pp. 15–23, 1999.

[38] ROSSET, S., ZHU, J., HASTIE, T. "Boosting as a Regularized Path to a Maximum Margin Classifier", *Journal of Machine Learning Research*, v. 5, pp. 941–973, 2004.

[39] KECMAN, V., HADZIC, I. "Support Vectors Selection by Linear Programming". In: *Proceedings of the International Joint Conference on Neural Networks*, pp. 193–198, 2000.

[40] PEDROSO, J. P., MURATA, N. "Support vector machines for linear programming: motivation and formulations", *BSIS Technical Report*, v. 99-2, 1999.

[41] PLATT, J. C. "Fast training of support vector machines using sequential minimal optimization". In: Schölkopf, B., Burges, C., Smola, A. J. (Eds.), *Advances in kernel methods: support vector learning*, pp. 185–208, Cambridge, MA, MIT Press, 1999.

[42] OSUNA, E. E., FREUND, R., GIROSI, F. *Support Vector Machines: Training and Applications*. Relatório técnico, Cambridge, MA, 1997.

[43] HUGHES, G. "On the mean accuracy of statistical pattern recognizers", *Information Theory, IEEE Transactions on*, v. 14, n. 1, pp. 55–63, 1968.

[44] MUCCIARDI, A. N., GOSE, E. E. "A Comparison of Seven Techniques for Choosing Subsets of Pattern Recognition Properties", *IEEE Transactions on Computers*, v. 20, pp. 1023–1031, 1971.

[45] LEOPOLD, E., KINDERMANN, J. "Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?" *Machine Learning*, v. 46, pp. 423–444, 2002.

[46] NIGAM, K., MCCALLUM, A., THRUN, S., et al. "Text Classification from Labeled and Unlabeled Documents using EM", *Machine Learning*, v. 39, pp. 103–134, 2000.

[47] YANG, Y., PEDERSEN, J. O. "A Comparative Study on Feature Selection in Text Categorization". In: *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pp. 412–420, San Francisco, CA, 1997. Morgan Kaufmann Publishers Inc.

[48] SWETS, D. L., WENG, J. J. "Efficient Content-Based Image Retrieval using Automatic Feature Selection". In: *Proceedings of the International Symposium on Computer Vision*, pp. 85–90, Washington, DC, 1995. IEEE Computer Society.

[49] RUI, Y., HUANG, T. S., CHANG, S.-F. "Image retrieval: Current techniques, promising directions and open issues", *Journal of Visual Communication and Image Representation*, v. 10, pp. 39–62, 1999.

[50] HASTIE, T., TIBSHIRANI, R., FRIEDMAN, J. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, Springer New York Inc., 2001.

[51] LEE, W., STOLFO, S. J., MOK, K. W. "Adaptive Intrusion Detection: a Data Mining Approach". In: *Artificial Intelligence Review*, v. 14, pp. 533–567, 2000.

[52] QUACKENBUSH, J. "Computational analysis of microarray data", *Nature Reviews Genetics*, v. 2, n. 6, pp. 418–27, 2001.

[53] DOAK, J. *An Evaluation of Feature Selection Methods and their Application to Computer Security*. Relatório Técnico CSE-92-18, University of California at Davis, 1992.

[54] FENG, T., XUEZHENG, F., YANQING, Z., et al. "Improving Feature Subset Selection Using a Genetic Algorithm for Microarray Gene Expression Data". In: Yen, G. G., Lucas, S. M., Fogel, G., et al. (Eds.), *Proceedings of the 2006 IEEE Congress on Evolutionary Computation*, pp. 2529–2534, Vancouver, BC, Canada, 2006. IEEE Press.

[55] JOHN, G. H., KOHAVI, R., PFLEGER, K. "Irrelevant Features and the Subset Selection Problem". In: *Proceedings of the Eleventh International Conference on Machine learning*, pp. 121–129, New Brunswick, NJ, 1994.

[56] GUYON, I. "Welcome and Introduction to the Problem of Feature/Variable Selection". In: *NIPS 2001 Workshop on Feature/Variable Selection*, 2001.

[57] YU, L., LIU, H. "Efficient Feature Selection via Analysis of Relevance and Redundancy", *Journal of Machine Learning Research*, v. 5, pp. 1205–1224, 2004.

[58] BLUM, A. L., LANGLEY, P. "Selection of relevant features and examples in machine learning", *Artificial Intelligence*, v. 97, n. 1-2, pp. 245–271, 1997.

[59] SIEDLECKI, W., SKLANSKY, J. "On Automatic Feature Selection", *International Journal of Pattern Recognition and Artificial Intelligence*, v. 2, n. 2, pp. 197–220, 1988.

[60] DASH, M., LIU, H. "Feature Selection for Classification", *Intelligent Data Analysis*, v. 1, pp. 131–156, 1997.

[61] JAIN, A., ZONGKER, D. "Feature selection: Evaluation, application, and small sample performance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 19, pp. 153–158, 1997.

[62] GUYON, I., ELISSEEFF, A. "An introduction to variable and feature selection", *Journal of Machine Learning Research*, v. 3, pp. 1157–1182, 2003.

[63] GOLUB, T. R., SLONIM, D. K., TAMAYO, P., et al. "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring", *Science*, v. 286(5439), pp. 531–537, 1999.

[64] BISHOP, C. *Neural networks for pattern recognition*. New York, Oxfor University Press, 1995.

[65] KIRA, K., RENDELL, L. A. "A Practical Approach to Feature Selection". In: *Proceedings of the Ninth International Workshop on Machine Learning*, ML '92, pp. 249–256, San Francisco, CA, 1992. Morgan Kaufmann Publishers Inc.

[66] WESTON, J., ELISSEEFF, A., SCHÖLKOPF, B., et al. "Use of the zero norm with linear models and kernel methods", *Journal of Machine Learning Research*, v. 3, pp. 1439–1461, 2003.

[67] TIBSHIRANI, R., HASTIE, T., NARASIMHAN, B., et al. "Diagnosis of multiple cancer types by shrunken centroids of gene expression", *Proceedings of the National Academy of Sciences*, v. 99, n. 10, pp. 6567–6572, 2002.

[68] KOHAVI, R., JOHN, G. H. "Wrappers for Feature Subset Selection", *Artificial Intelligence*, v. 97, n. 1-2, pp. 273–324, 1997.

[69] GUYON, I., WESTON, J., BARNHILL, S., et al. "Gene Selection for cancer classification using support vector machines", *Machine Leaning*, v. 46, pp. 389–422, 2002.

[70] VILLELA, S. M., FONSECA NETO, R., LEITE, S. C., et al. "Seleção de Características utilizando Busca Ordenada e um Classificador de Larga Margem". In: *X CBIC - Congresso Brasileiro de Inteligência Computacional*, Fortaleza, CE, 2011.

[71] HART, P., NILSSON, N., RAPHAEL, B. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Transactions on Systems Science and Cybernetics*, v. 4, n. 2, pp. 100–107, 1968.

[72] ASUNCION, A., NEWMAN, D. J. *UCI Machine Learning Repository*. Irvine, CA: University of California, School of Information and Computer Science, 2007. Disponível em: <`http://www.ics.uci.edu/~mlearn/MLRepository.html`>.

[73] ALON, U., BARKAI, N., NOTTERMAN, D. A., et al. "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays", *Proceedings of the National Academy of Sciences of the United States of America*, v. 96, n. 12, pp. 6745–6750, 1999.

[74] SINGH, D., FEBBO, P. G., ROSS, K., et al. "Gene expression correlates of clinical prostate cancer behavior", *Cancer Cell*, v. 1, n. 2, pp. 203–209, 2002.

[75] PEDROSO, J. P., MURATA, N. "Support vector machines with different norms: motivation, formulations and results", *Pattern Recognition Letters*, v. 22, n. 12, pp. 1263–1272, 2001.

[76] MOSTELLER, F., TUKEY, J. W. "Data analysis, including statistics". In: Lindzey, G., Aronson, E. (Eds.), *Handbook of Social Psychology, Vol. 2*. Addison-Wesley, 1968.

[77] GEISSER, S. "The Predictive Sample Reuse Method with Applications", *Journal of the American Statistical Association*, v. 70, n. 350, 1975.

[78] WAHBA, G., WOLD, S. "A completely automatic french curve: Fitting spline functions by cross-validation", *Communications in Statistics*, v. 4, n. 1, 1975.

[79] BROWNE, M. "Cross-Validation Methods", *Journal of Mathematical Psychology*, v. 44, n. 1, pp. 108–132, 2000.

[80] KOHAVI, R. "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, pp. 1137–1143, San Francisco, CA, 1995. Morgan Kaufmann Publishers Inc.

[81] MCLACHLAN, G. J., DO, K. A., AMBROISE, C. *Analyzing microarray gene expression data*. Wiley series in probability and statistics. Hoboken, New Jersey, Wiley-Interscience, 2004.

[82] TIBSHIRANI, R., HASTIE, T., NARASIMHAN, B., et al. "Class prediction by nearest shrunken centroids, with applications to DNA microarrays", *Statistical Science*, v. 18(1), pp. 104–117, 2003.

[83] FONSECA NETO, R., BORGES, C. C. H., LEITE. "An online training method based in support vectors for regression problems". In: *XXIX CILAMCE*, Macéio, AL, 2008.

[84] WU, K.-P., WANG, S.-D. "Choosing the kernel parameters for support vector machines by the inter-cluster distance in the feature space", *Pattern Recogn.*, v. 42, pp. 710–717, 2009.

[85] DEBNATH, R., TAKAHASHI, H. "An efficient method for tuning kernel parameter of the support vector machine". In: *IEEE International Symposium on Communications and Information Technology, 2004. ISCIT 2004*, v. 2, 2004.

[86] SCHITTKOWSKI, K. "Optimal parameter selection in support vector machines", *Journal of Industrial and Management Optimization*, v. 1, 2005.