Instituto de Ciências Matemáticas e de Computação

ISSN - 0103-2569

Introdução às Máquinas de Vetores Suporte (*Support Vector Machines*)

Ana Carolina Lorena
André C. P. L. F. de Carvaho

Nº 192

RELATÓRIOS TÉCNICOS DO ICMC

São Carlos
Abril/2003

Introdução às Máquinas de Vetores Suporte
(*Support Vector Machines*)*

Ana Carolina Lorena
André C. P. L. F. de Carvalho

Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
Departamento de Ciências de Computação e Estatística
Laboratório de Inteligência Computacional
Caixa Postal 668, 13560-970 - São Carlos, SP, Brasil
e-mail: {aclorena, andre}@icmc.sc.usp.br

Resumo. Este relatório tem por objetivo apresentar uma introdução às Máquinas de Vetores Suporte (*Support Vector Machines* - SVMs), técnica de Aprendizado de Máquina que vem recebendo grande atenção nos últimos anos. As SVMs vêm sendo utilizadas em diversas tarefas de reconhecimento de padrões, obtendo resultados superiores aos alcançados por técnicas similares em várias aplicações.

Palavras-Chave: Máquinas de Vetores Suporte (*Support Vector Machines*), Aprendizado de Máquina supervisionado.

Abril 2003

*Trabalho realizado com auxílio da FAPESP.

Este documento foi preparado com o formatador de textos LaTeX. O sistema de citações de referências bibliográficas utiliza o padrão *Apalike* do sistema bibTeX.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Capítulo 1

Introdução

As Máquinas de Vetores Suporte (*Support Vector Machines* - SVMs) constituem uma técnica embasada na Teoria de Aprendizado Estatístico (Vapnik, 1995) que vem recebendo grande atenção nos últimos anos (Hearst et al., 1998; Cristianini and Shawe-Taylor, 2000).

Os resultados da aplicação desta técnica são comparáveis aos obtidos por outros algoritmos de aprendizado, como as Redes Neurais Artificiais (RNAs) (Haykin, 1999), e em algumas tarefas têm se mostrado superiores, tal como na detecção de faces em imagens (Hearst et al., 1998), na categorização de textos (Hearst et al., 1998) e em aplicações em Bioinformática (Zien et al., 2000).

Este relatório apresenta uma introdução às SVMs. Iniciando as descrições a respeito dessa técnica, a Seção 1.1 apresenta uma breve descrição de conceitos de Aprendizado de Máquina (AM), campo de estudo em Inteligência Computacional do qual as SVMs fazem parte. A Seção 1.2 lista algumas das principais características das SVMs que tornam seu uso atrativo. A organização deste relatório é apresentada na Seção 1.3.

1.1 Aprendizado de Máquina

Aprendizado de Máquina (AM) é um campo de pesquisa da Inteligência Computacional que estuda o desenvolvimento de métodos capazes de extrair conceitos (conhecimento) a partir de amostras de dados (Mitchell, 1997).

Em geral, os diversos algoritmos de AM são utilizados de forma a gerar classificadores para um conjunto de exemplos. Por classificação entende-se o processo de atribuir, a uma determinada informação, o rótulo da classe[1] a qual ela pertence (Russel and Norvig, 1995). Portanto, as técnicas de AM são empregadas na indução (a partir de um conjunto

[1] A classe de um exemplo (instância) descreve o fenômeno de interesse, ou seja, o que se deseja aprender e fazer previsões (Baranauskas and Monard, 2000).

de treinamento) de um classificador, que deve ser capaz (idealmente) de prever a classe de instâncias quaisquer do domínio em que ele foi treinado.

Três paradigmas podem ser utilizados na geração de um preditor por meio de técnicas de AM: superviosionado, não-supervisionado e por reforço (Haykin, 1999). A escolha de um *paradigma de aprendizado* determina a maneira como o algoritmo de AM se relaciona com seu meio ambiente, ou seja, o modo como ocorrerá o seu aprendizado por meio de um conjunto de dados.

No paradigma de *aprendizado supervisionado* tem-se a figura de um "professor externo", o qual apresenta um conhecimento do ambiente representado por conjuntos de exemplos na forma entrada- saída. Neste caso, o algoritmo de AM é treinado a partir de conjuntos de exemplos rotulados com o objetivo de aprender uma função desejada.

No paradigma de aprendizado *não-supervisionado* não há a presença de um professor, ou seja, não existem instâncias rotuladas da função a ser aprendida. O algoritmo de AM aprende a representar (ou agrupar) as entradas submetidas segundo uma medida de qualidade.

O último paradigma é o *por reforço*, em que o aprendizado se dá por meio de recompensas ou não ao indutor, dependendo de seu desempenho em aproximar a função desejada.

Este relatório foca a utilização das SVMs em tarefas de classificação por meio de treinamento supervisionado, embora seu uso também seja reportado em problemas de regressão (Tay and Cao, 2001; Herbrich et al., 1999) e "clusterização" (aprendizado não supervisionado) (Ben-Dor et al., 2000).

No desenvolvimento de técnicas de AM, diversos pesquisadores se inspiraram nos sistemas biológicos. Seguindo essa linha tem-se as Redes Neurais Artificiais, cujos conceitos são baseados nos mecanismos de aprendizado cerebrais, e os Algoritmos Genéticos, que têm seus conceitos inspirados no processo de evolução natural e na genética. No campo de aprendizado simbólico, tem-se as Árvores de Decisão. Inspirado nos processos cognitivos, houve o desenvolvimento dos Sistemas de Raciocínio Baseado em Casos. As SVMs, por sua vez, fundamentam-se em Teorias Estatísticas. Algumas de suas características são discutidas a seguir.

1.2 Características das SVMs

Algumas das principais características das SVMs que tornam seu uso atrativo são (Smola et al., 1999b):

- **Boa capacidade de generalização:** os classificadores gerados por uma SVM em

geral alcançam bons resultados de generalização. A capacidade de generalização de um classificador é medida por sua eficiência na classificação de dados que não pertençam ao conjunto utilizado em seu treinamento. Na geração de preditores por SVMs, portanto, é evitado o *overfitting*, situação na qual o preditor se torna muito especializado no conjunto de treinamento, obtendo baixo desempenho quando confrontado com novos padrões.

- **Robustez em grandes dimensões:** as SVMs são robustas diante de objetos de grandes dimensões, como, por exemplo, imagens. Comumente há a ocorrência de *overfitting* nos classificadores gerados por outros métodos inteligentes sobre esses tipos de dados.

- **Convexidade da função objetivo:** a aplicação das SVMs implica na otimização de uma função quadrática, que possui apenas um mínimo global. Esta é uma vantagem sobre, por exemplo, as Redes Neurais Artificiais, em que há a presença de mínimos locais na função objetivo a ser minimizada.

- **Teoria bem definida:** as SVMs possuem uma base teórica bem estabelecida dentro da Matemática e Estatística.

Entre as características citadas, o destaque das SVMs está em sua capacidade de generalização. Estes resultados foram apresentados por Vapnik e Chernovenkis através da Teoria de Aprendizado Estatístico, proposta por estes autores nas décadas de 60 e 70 (Vapnik, 1995). As SVMs surgiram como resultado direto do emprego dos princípios apresentados nestes estudos. Apesar de sua teoria ser relativamente antiga, as primeiras aplicações práticas das SVMs são recentes, e datam da década de 90 (Hearst et al., 1998).

1.3 Organização do Relatório

Como as SVMs são baseadas na Teoria de Aprendizado Estatístico, o Capítulo 2 apresenta os resultados principais desta teoria e como as SVMs fazem uso de seus princípios.

A seguir, o Capítulo 3 apresenta a forma mais simples de SVMs, que realizam a classificação de conjuntos linearmente separáveis.

O Capítulo 4 estende os resultados apresentados para a classificação linear de conjuntos mais gerais.

O Capítulo 5 apresenta as SVMs não lineares.

Uma vez que as SVMs são propostas para problemas de classificação binária (envolvendo apenas duas classes), o Capítulo 6 mostra como estas podem ser utilizadas em problemas que envolvam a existência de mais classes (multiclasses).

O Capítulo 7 lista abordagens para o treinamento das SVMs, assim como algumas implementações disponíveis.

O Capítulo 8 apresenta algumas aplicações de SVMs.

O Capítulo 9 aponta as principais conclusões deste estudo.

Finalizando, o Apêndice A apresenta uma introdução simplificada à Teoria de Otimização de funções, visando facilitar a compreensão de alguns conceitos Matemáticos apresentados no decorrer deste relatório.

Capítulo 2

A Teoria de Aprendizado Estatístico

Seja f um classificador, isto é, um mapeamento de um conjunto de padrões $\mathbf{x_i} \in \Re^m$ para o espaço de classes, e F o conjunto de todos classificadores que um determinado algoritmo de AM pode gerar[1]. Este algoritmo, durante o aprendizado[2], utiliza um conjunto de treinamento S, composto de n pares $(\mathbf{x_i}, y_i)$, em que y_i representa a classe do padrão $\mathbf{x_i}$, para gerar um classificador particular $f' \in F$.

Considere por exemplo[3] o conjunto de treinamento da Figura 2.1. Deve-se encontrar um classificador (ou função) que separe as instâncias de treinamento das classes "círculo" e "triângulo". As funções ou hipóteses consideradas são ilustradas na figura através das bordas, também denominadas fronteiras de decisão, traçadas entre as classes. Na imagem 2.1c, tem-se uma hipótese que classifica corretamente todos os exemplos do conjunto de treinamento. A função utilizada, porém, tem grandes chances de cometer erros quando confrontada com exemplos de teste distintos dos de treinamento, pois é muito específica para os últimos. Esse caso representa a ocorrência de *overfitting*, em que considera-se que o algoritmo "memorizou" os dados do conjunto de treinamento.

Um outro modelo poderia desconsiderar pontos de classes opostas muito próximos entre si, pois esses dados são pouco confiáveis (mesmo uma pequena quantidade de ruídos pode levar a uma mudança do valor de sua classe). A ilustração 2.1a representa esta alternativa, em que se utiliza uma fronteira de decisão linear. A nova hipótese considerada, porém, comete muitos erros, mesmo para casos considerados simples. Pode-se considerar que houve a ocorrência de *underfitting*, pois o classificador não é capaz de se ajustar até mesmo às instâncias de treinamento.

Um compromisso entre as duas funções apresentadas anteriormente é representado em

[1] Váriáveis vetoriais serão denotadas em negrito neste relatório.

[2] As descrições sendo realizadas dizem respeito ao paradigma de aprendizado supervisionado, em que os exemplos são rotulados.

[3] Baseado em exemplo fornecido em (Smola and Schölkopf, 2002).

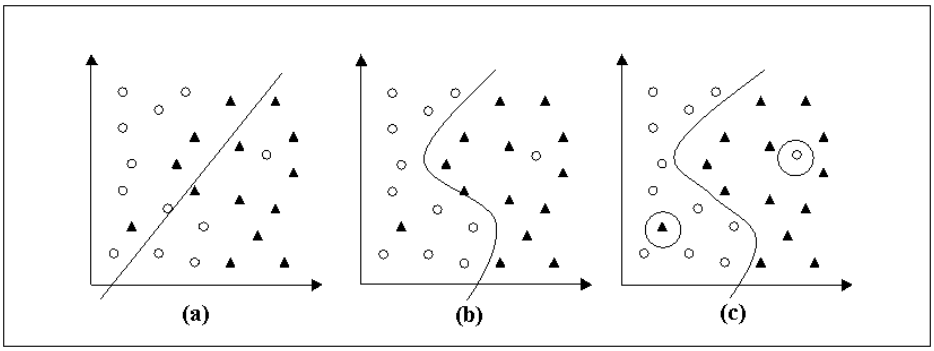

Figura 2.1: Exemplo de conjunto de dados de treinamento pertencentes a duas classes classificado segundo três diferentes hipóteses.

2.1b, em que o preditor tem complexidade intermediária e classifica corretamente grande parte dos dados.

A Teoria de Aprendizado Estatístico (TAE) visa estabelecer condições matemáticas que permitam a escolha de um classificador f' com bom desempenho para os conjuntos de treinamento e teste. Ou seja, busca-se uma função f' capaz de classificar os dados de treinamento da forma mais correta possível, sem dar atenção exacerbada a qualquer ponto individual do mesmo. A próximas Seções descrevem como se dá tal escolha.

2.1 Considerações sobre a Escolha do Classificador

Na escolha de uma função particular f', é natural considerar a função que produz o menor erro durante o treinamento, ou seja, aquela que possui maior habilidade de classificar corretamente os padrões do conjunto de treinamento S. Seja o risco empírico de f ($R_{emp}(f)$), definido pela Equação 2.1, a porcentagem de classificações incorretas obtidas em S, em que $c(.)$ é uma função de custo relacionando a previsão $f(\mathbf{x}_i)$ com a saída desejada y_i (Müller et al., 2001). Um tipo de função de custo é a "perda 0/1", definida por $c(f(\mathbf{x}), y) = 1$ se $yf(\mathbf{x}) < 0$ e 0 caso contrário (Müller et al., 2001).

O processo de busca por uma função f' que apresente menor R_{emp} é denominado *Minimização do Risco Empírico*.

$$R_{emp}(f) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}c(f(\mathbf{x}_i), y_i) \tag{2.1}$$

Supõe-se que os exemplos do domínio em que está ocorrendo o aprendizado (incluindo os de treinamento) são gerados independentemente e identicamente distribuídos (i.i.d.) de acordo com uma distribuição de probabilidade P. O desempenho de generalização de

um classificador f pode então ser definido como a probabilidade de que f cometa erro na classificação de um novo exemplo gerado segundo P. A partir desta afirmação, define-se o *Risco Funcional* $R(f) = P(f(X) \neq Y)$ (Equação 2.2), que quantifica a capacidade de generalização de f (se $R(f)$ for igual a 0, pode-se afirmar que f generaliza de forma "perfeita").

$$R(f) = \int \frac{1}{2} c(f(\mathbf{x}), y) dP(\mathbf{x}, y) \tag{2.2}$$

Durante o treinamento, $R_{emp}(f)$ é facilmente observável. O mesmo não pode ser afirmado sobre $R(f)$, pois em geral a distribuição de probabilidade P é desconhecida. A maioria dos algoritmos de aprendizado busca a minimização de $R_{emp}(f)$, esperando que esta leve a um menor $R(f)$. Portanto, na escolha de uma função f, geralmente se opta por uma função $\hat{f}$ tal que $R_{emp}\left(\hat{f}\right) = \min_{f \in F} R_{emp}(f)$. Porém, também é desejável que o classificador possua uma boa capacidade de generalização, ou seja, que se tenha f^* tal que $R(f^*) = \min_{f \in F} R(f)$.

Em geral, $\hat{f}$ leva a uma boa aproximação de f^*. Contudo, nem sempre isto é verdade. Considere, por exemplo, um estimador que memoriza todos os dados de treinamento e gera classificações aleatórias para outros exemplos (Smola et al., 1999b). Este classificador possui um risco empírico nulo, porém seu risco funcional é igual a 1/2.

A TAE provê diversos limites no risco funcional de uma função, os quais podem ser então utilizados na escolha do classificador. A próxima Seção relaciona alguns dos principais limites sobre os quais as SVMs se baseiam.

2.2 Limites no Risco Funcional

Seja S um conjunto de treinamento em que cada exemplo $\mathbf{x}_i$ pertencente ao espaço $\Re^m$ e os rótulos correspondentes y_i assumem valores -1 ou +1. A partir de um processo de indução, o objetivo é encontrar uma função $g : \Re^m \rightarrow \{-1, +1\}$ capaz de predizer a classe de novos pontos $(\mathbf{x}, y)$ de forma precisa.

Uma forma de realizar esta tarefa é utilizar uma função do tipo sinal, representada na Equação 2.3.

$$g(\mathbf{x}) = g(f(\mathbf{x})) = \begin{cases} +1 \ \ se \ f(\mathbf{x}) > 0 \\ -1 \ \ se \ f(\mathbf{x}) < 0 \end{cases} \tag{2.3}$$

e $g(\mathbf{x})$ é desconhecido se $f(\mathbf{x}) = 0$.

Esta função recebe a saída de outra função $f(\mathbf{x})$, a qual define então uma fronteira de separação entre os dados. Um exemplo é ilustrado na Figura 2.2.

Os limites no risco funcional para funções sinal relacionam o número de exemplos de

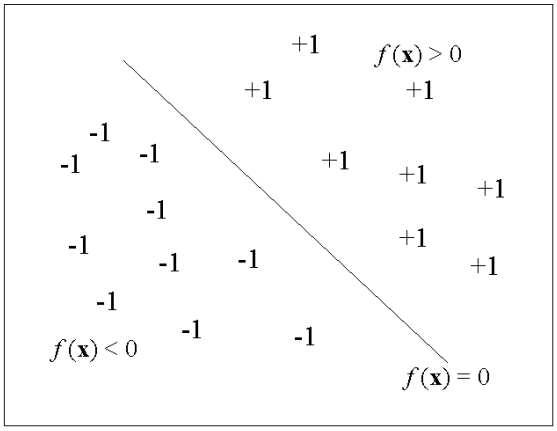

Figura 2.2: Exemplo de função sinal.

treinamento, o risco empírico obtido neste conjunto e a complexidade do espaço de hipóteses. Esta complexidade é medida através do conceito de dimensão Vapnik-Chervonenkis (VC), definida a seguir.

2.2.1 A Dimensão VC

Dado um conjunto de funções sinal G, sua dimensão VC é definida como o tamanho do maior conjunto de pontos que pode ser particionado arbitrariamente pelas funções contidas em G (Smola et al., 1999b).

Para facilitar a compreensão deste conceito, pode-se reescrever a Equação 2.3 da forma representada em 2.4, em que a função $g(\mathbf{x})$ particiona o espaço de entradas em dois subconjuntos disjuntos S_0 e S_1. Funções deste tipo também são denominadas dicotomias (Haykin, 1999).

$$g(\mathbf{x}) = \begin{cases} +1 \ \ se \ \mathbf{x} \in S_0 \\ -1 \ \ se \ \mathbf{x} \in S_1 \end{cases} \tag{2.4}$$

Seja $\Delta_G(S)$ o número de dicotomias que o algoritmo de aprendizado tem capacidade de induzir sobre S. Diz-se que S é "fragmentado" por G se $\Delta_G(S) = 2^{|S|}$ ($|.|$ representa a cardinalidade, ou tamanho, de um conjunto), isto é, se as funções contidas em G são capazes de induzir todas as possíveis dicotomias sobre S.

A dimensão VC de um conjunto de dicotomias G é então definida como a cardinalidade do maior conjunto S que é fragmentado por G, ou seja, o maior N tal que $\Delta_G(S) = 2^N$, em que $N = |S|$. Caso este valor não possa ser estimado, N assume o valor ∞.

Logo, um alto valor de dimensão VC indica uma grande complexidade das funções de decisão contidas em G, já que essas se tornam capazes de fragmentar conjuntos de dados também complexos.

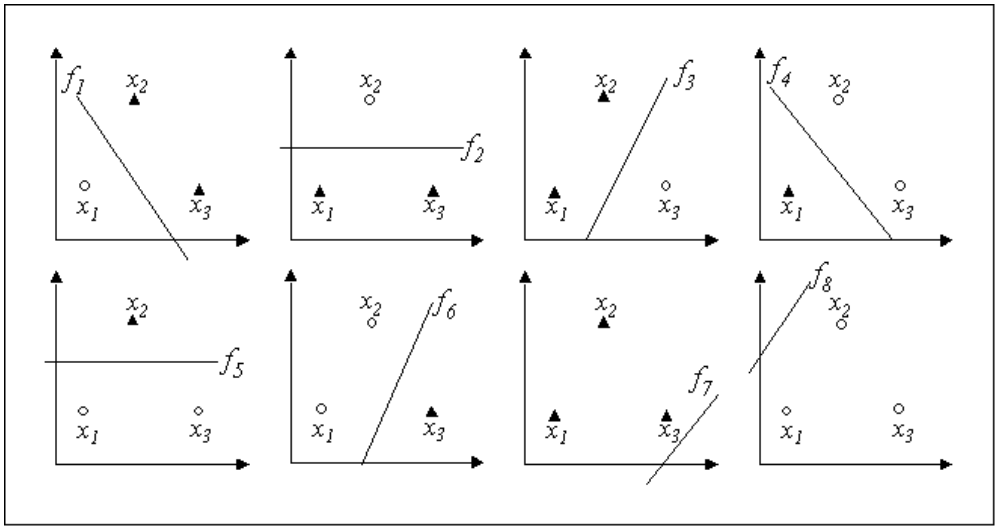

Figura 2.3: Exemplo de distribuição de três pontos no estado bidimensional. Verifica-se que para todas as dicotomias possíveis, é possível traçar retas que separem as classes "círculo" e "triângulo".

O Exemplo 1, extraído de (Haykin, 1999), ilustra de forma mais clara os conceitos apresentados.

Exemplo 1: Seja G' o conjunto de funções sinal com fronteira linear, como representado na Equação 2.5.

$$G' : g(\mathbf{x}) = \text{sgn}(f(\mathbf{x}) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b) \tag{2.5}$$

em que $\mathbf{x}$ representa uma entrada m-dimensional, $\mathbf{w}$ o vetor de pesos e b o *bias*.

A dimensão VC do conjunto de funções G' 2.5 é dada por 2.6.

$$V(G') = m + 1 \tag{2.6}$$

em que m é a dimensão de $\mathbf{x}$.

Este resultado pode ser analisado através da observação das Figuras 2.3 e 2.4.

Na Figura 2.3, há três pontos bidimensionais, o que gera oito possíveis combinações binárias dos dados, ou seja, oito dicotomias. Um número de oito retas é suficiente para fragmentar estes dados em todas classificações binárias admissíveis. As classificações possíveis e retas utilizadas em sua fragmentação são sumarizadas na Tabela 2.1.

Na Figura 2.4, tem-se os pontos bidimensionais $\mathbf{x_1}$, $\mathbf{x_2}$, $\mathbf{x_3}$ e $\mathbf{x_4}$. Uma das possíveis distribuições considerada é aquela em que $\mathbf{x_1}$ e $\mathbf{x_4}$ pertencem à classe "círculo" e $\mathbf{x_2}$ e $\mathbf{x_3}$ são da classe "triângulo". Neste caso não é possível separar os dados com o uso de uma reta. É necessária uma função de maior complexidade para realização desta tarefa.

x_1	x_2	x_3	**Função**
○	▲	▲	f_1
▲	○	▲	f_2
▲	▲	○	f_3
▲	○	○	f_4
○	▲	○	f_5
○	○	▲	f_6
▲	▲	▲	f_7
○	○	○	f_8

Tabela 2.1: Ilustração esquemática do exemplo apresentado na Figura 2.3.

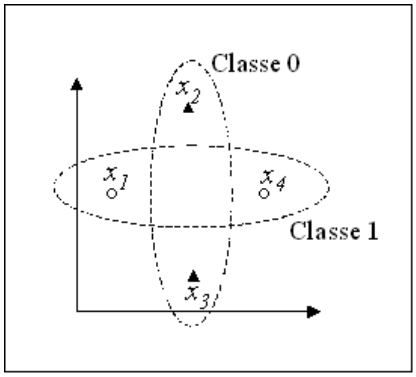

Figura 2.4: Distribuição de quatro pontos no espaço bidimensional. Para este caso, verifica-se que não é possível separar as classes por meio de uma reta.

A dimensão VC da Equação 2.5 no caso bidimensional é, portanto, igual a 3, em conformidade com a Equação 2.6.

2.2.2 Limites Baseados na Dimensão VC

Os teoremas seguintes provêem limites no risco funcional ($R(.)$) de uma função baseados na dimensão VC do espaço de hipóteses do qual o classificador é extraído (Smola et al., 1999b).

Teorema 2.1 (Limite superior) *Seja G um conjunto de funções de decisão mapeando $\Re^m$ a $\{-1,+1\}$ com dimensão VC h. Para qualquer distribuição de probabilidade P em $\Re^m x\{-1,+1\}$, com probabilidade de ao menos $1-\delta$ sobre n exemplos e para qualquer*

hipótese g em G o risco funcional é limitado por

$$R(g) \leq R_{emp}(g) + \sqrt{\frac{c}{n}\left(h + \ln\left(\frac{1}{\delta}\right)\right)} \tag{2.7}$$

em que c é uma constante universal. Se $\hat{g} \in G$ minimiza o risco empírico ($R_{emp}(.)$), então com probabilidade $1 - \delta$

$$R(\hat{g}) \leq \inf_{g \in G} R(g) + \sqrt{\frac{c}{n}\left(h + \ln\left(\frac{1}{\delta}\right)\right)} \tag{2.8}$$

Teorema 2.2 (Limite inferior) *Seja G um conjunto de funções de decisão mapeando $\Re^m$ a $\{-1, +1\}$ com dimensão VC finita $h \geq 1$. Para qualquer algoritmo de aprendizado há distribuições tal que com probabilidade de ao menos $1 - \delta$ sobre n exemplos, o risco funcional da hipótese $g \in G$ satisfaz*

$$R(g) \geq \inf_{g' \in G} R(g') + \sqrt{\frac{c}{n}\left(h + \ln\left(\frac{1}{\delta}\right)\right)} \tag{2.9}$$

em que c é uma constante universal.

O termo mais à direita das Equações 2.7, 2.8 e 2.9 é também denominado termo de capacidade do conjunto de funções G.

Através da observação das equações apresentadas nos Teoremas 2.1 e 2.2, as seguintes asserções podem ser realizadas:

- O risco funcional de uma função g é minimizado se o número de observações n do conjunto de treinamento for suficientemente grande. Quanto maior a amostra de dados apresentada ao algoritmo de aprendizado, mais a minimização do risco empírico se aproxima do risco mínimo que o espaço de funções G pode alcançar;

- O risco médio de g é minimizado se a dimensão VC de G for suficientemente pequena. Ou seja, quanto menor a dimensão VC de uma função $\hat{g}$ escolhida pela Minimização do Risco Empírico, maior sua capacidade de generalização.

A contribuição principal dos teoremas apresentados está em afirmar a importância de se controlar o termo de capacidade apresentado nas equações 2.7 a 2.9.

Como os limites apresentados dizem respeito a uma classe de funções G e não simplesmente a escolhas de funções g, introduz-se uma "estrutura" em G e realiza-se a minimização dos limites sobre escolhas de estruturas. Este princípio é denominado *Minimização do Risco Estrutural* (Smola and Schölkopf, 2002).

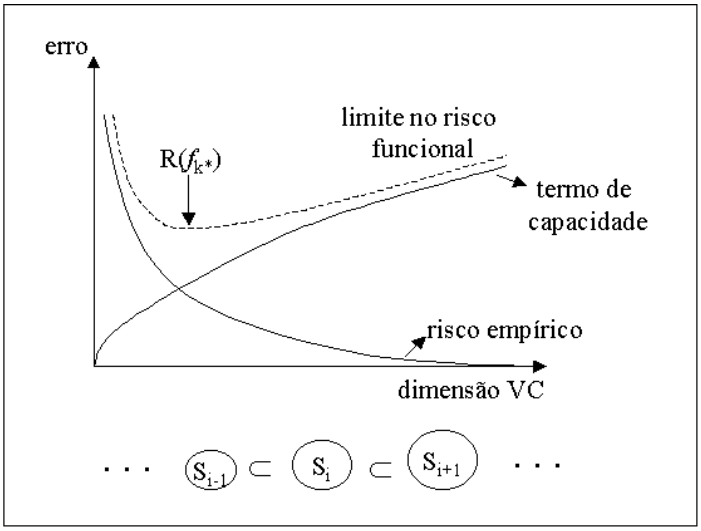

Figura 2.5: Princípio de Minimização do Erro Estrutural (Smola and Schölkopf, 2002).

Considera-se conjuntos de estruturas S_i, com complexidade crescente ($\ldots S_{i-1} \subset S_i \subset S_{i+1} \ldots$). Como as estruturas são maiores à medida que i cresce, a capacidade das mesmas também é maior com o crescimento deste índice.

Para cada S_k, seja $\hat{f}_k$ o classificador com menor Risco Empírico e f_k^* o que possui menor Risco Funcional. A medida que k cresce, o risco empírico diminui, já que a complexidade do conjunto de classificadores (ou seja, das fronteiras de decisão a serem geradas) é maior. Porém, o termo de capacidade aumenta com k. Como resultado, os limites em $R(\hat{f})$ fornecidos nos Teoremas 2.1 e 2.2 inicialmente decrescem com o aumento de k, e depois crescem. Logo, deve haver um valor ótimo k^* tal que se obtém um erro de generalização mínimo, conforme ilustrado na Figura 2.5. A escolha da função $\hat{f}_{k^*}$ constitui o princípio da Minimização do Risco Estrutural.

Embora os Teoremas 2.1 e 2.2 caracterizem o risco funcional de uma forma razoavelmente completa, na prática esta caracterização tem alguns problemas. Em primeiro lugar, computar os limites apresentados não é uma tarefa fácil (Müller et al., 2001). Podem haver problemas também quando a dimensão VC de uma classe de funções é desconhecida ou infinita.

Existem diversos outros limites reportados na literatura, assim como outros tipos de medida de complexidade de uma função. Porém, grande parte delas possuem os mesmos problemas discutidos.

Apesar dessas questões, os limites oferecidos fornecem uma base teórica importante em relação à natureza dos problemas de aprendizado e em direção à formulação de estratégias para a geração de classificadores (Müller et al., 2001).

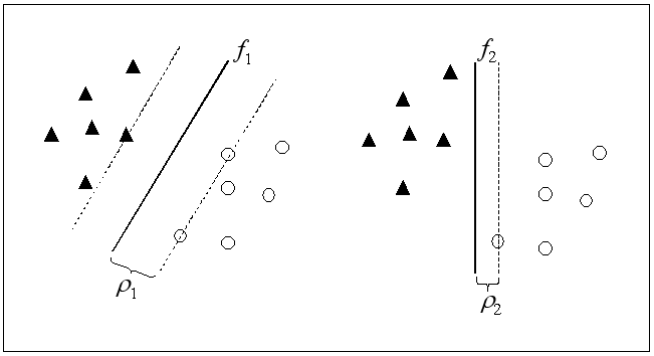

Figura 2.6: Exemplos de hiperplanos separadores e margens correspondentes.

2.2.3 A Dimensão VC e o Conceito de Margem

Seja $g(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$ uma função sinal em que $f(\mathbf{x})$ é um hiperplano, isto é, $f(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$. Existem resultados em relação a este tipo de função relacionando a dimensão VC ao conceito de margem do classificador (Müller et al., 2001).

A margem de um classificador é definida como a menor distância entre os exemplos do conjunto de treinamento e o hiperplano utilizado na separação desses dados em classes. A Figura 2.6 ilustra um conjunto de dados e dois possíveis hiperplanos separadores com suas respectivas margens.

O Teorema 2.3 relaciona a dimensão VC de funções de decisão lineares com a margem do classificador (Smola et al., 1999b).

Teorema 2.3 *Seja $X_0 \subset \Re^m$ o conjunto de entradas com norma menor que $R > 0$ ($\|\mathbf{x}_i\| \leq R$, para todo $\mathbf{x}_i \in X_0$) e F o conjunto de funções lineares definidas em X_0 e satisfazendo $\|f(\mathbf{x})\| \geq \rho$, em que ρ é a margem do classificador.*

$$F = \{\mathbf{x} \rightarrow \mathbf{w} \cdot \mathbf{x} | \, \|\mathbf{w}\| \leq 1, \, \mathbf{x} \in X_0\} \tag{2.10}$$

Considerando G o conjunto de funções sinal obtidas à partir de $G = \text{sgn}(F)$ e h a dimensão VC de G, tem-se o resultado representado pela Equação 2.11.

$$h \leq \min\left\{\frac{R^2}{\rho^2}, m\right\} + 1 \tag{2.11}$$

Pode-se observar através deste resultado que a dimensão VC de um conjunto de funções limiares $G = \text{sgn}(F)$ com F linear pode ser menor que o valor $m+1$ calculado no Exemplo 1, em que não se levava em conta a margem ρ (Smola et al., 1999b). Este teorema, portanto, completa a noção de dimensão VC para fronteiras lineares. Segundo o mesmo

teorema, a margem do classificador linear e a dimensão VC do espaço de hipóteses do qual este é extraído se relacionam de forma indiretamente proporcional, ou seja, quanto maior a margem de um classificador menor sua dimensão VC. Porém, o limite apresentado é restrito, pois através dele se requer que todos exemplos, inclusive os de teste, estejam afastados do hiperplano de uma margem ρ.

O Teorema 2.4 apresenta limites no risco funcional de funções do tipo sinal com fronteiras de decisão lineares em função da margem unicamente (Smola and Schölkopf, 2002).

Teorema 2.4 *Definindo a margem ρ de um classificador f como*

$$\rho = \min_i y_i f(\mathbf{x}_i) \tag{2.12}$$

, seja o erro marginal de f ($R_\rho(f)$) a proporção de exemplos de treinamento que têm margem menor que ρ.

$$R_\rho(f) = \frac{1}{n} \sum_{i=1}^{n} |y_i f(\mathbf{x}_i) < \rho| \tag{2.13}$$

Seja G o conjunto de funções $g(\mathbf{x}) = \operatorname{sgn}(f(\mathbf{x})) = \operatorname{sgn}(\mathbf{w} \cdot \mathbf{x})$ com $\|\mathbf{w}\| \leq \Lambda$ e $\|\mathbf{x}\| \leq R$, para algum $R, \Lambda > 0$. Seja $\rho > 0$. Para todas distribuições P gerando os dados, com probabilidade de ao menos $1 - \delta$ sobre n exemlos, e para qualquer $\rho > 0$ e $\delta \in (0, 1)$, a probabilidade de um ponto de teste amostrado independentemente segundo P ser classificado incorretamente é limitado superiormente por

$$R_\rho(g) + \sqrt{\frac{c}{n}\left(\frac{R^2\Lambda^2}{\rho^2}\ln^2 n + \ln\left(\frac{1}{\rho}\right)\right)} \tag{2.14}$$

em que c é uma constante universal.

Deve-se observar que o limite apresentado no Teorema 2.4 não se refere a um único classificador, mas a um conjunto de preditores treinados em diferentes conjuntos de dados gerados com a mesma distribuição de probabilidade P. Porém, sua análise fornece resultados interessantes. Pode-se verificar que o risco funcional é limitado pela soma do erro marginal a um termo de capacidade, que tende a 0 quando o número de exemplos n tende a infinito. Este termo pode ser minimizado mantendo-se R e Λ pequenos e maximizando-se a margem ρ. Fixando R e Λ, o termo de maior importância torna-se a margem ρ. Maximizar ρ leva a um termo de capacidade pequeno. Porém, o erro marginal $R_\rho(\cdot)$ torna-se maior, pois é mais difícil obedecer a restrição de todos dados de treinamento estarem distantes de uma margem maior do hiperplano separador. Um baixo valor de ρ, por sua vez, leva a um erro marginal menor, porém aumenta o termo de capacidade. Deve-se buscar, portanto, o hiperplano que tenha margem ρ alta e cometa poucos erros marginais, minimizando-se assim o erro sobre os dados de teste e de treinamento, respectivamente.

O hiperplano que possui esta margem é denominado ótimo.

O hiperplano ótimo, que procura maximizar a margem de separação entre os dados, também possui duas propriedades interessantes: robustez em relação aos padrões e robustez em relação aos parâmetros (Smola and Schölkopf, 2002).

Na robustez em relação aos padrões, tem-se que dado um exemplo $\mathbf{x}$ longe da borda de decisão formada por $f(\mathbf{x}) = 0$, ocasionais perturbações em $\mathbf{x}$ não levarão a uma classificação incorreta $g(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$.

Para o caso da robustez em relação aos parâmetros, se a margem se separação é grande, uma pequena perturbação nos parâmetros de f não afeta a classificação dos dados.

2.3 Considerações Finais

Este Capítulo descreveu a Teoria de Aprendizado Estatístico, proposta por Vapnik e Chervonenkis no final da década de 60 (Vapnik, 1995). Esta teoria apresenta condições matemáticas para a escolha de uma função (hipótese) que separe os dados de treinamento de forma a efetivamente maximizar a generalização do classificador obtido. Para isto, busca-se um compromisso entre a minimização do erro de treinamento e a complexidade da função escolhida para separação dos dados em classes. Essas condições são alcançadas através do controle da dimensão VC do conjunto de funções do qual o classificador deve ser extraído. A dimensão VC mede a complexidade das hipóteses examinadas pelo algoritmo na busca por uma solução.

Apresentou-se também uma introdução à forma como esses princípios pode ser utilizados na geração de classificadores lineares, através da maximização da margem de separação entre os dados de treinamento e a fronteira de decisão induzida para separação destes em classes.

Capítulo 3

SVMs com Margens Rígidas

As SVMs são originalmente utilizadas para classificação de dados em duas classes, ou seja, na geração de dicotomias. Nas considerações realizadas nos Capítulos 3 a 5, supõe-se que se deseja classificar objetos m-dimensionais $\mathbf{x} = (x_1, x_2, \ldots, x_m)$ nas classes +1 e -1. Portanto, o conjunto de treinamento consiste de n observações $\mathbf{x_i} \in \Re^m$, com suas respectivas classificações binárias.

Este Capítulo apresenta como as SVMs podem ser utilizadas para classificação de conjuntos de treinamento linearmente separáveis, definidos a seguir. Estas SVMs são também denominadas *SVMs com margens rígidas*.

3.1 Conjuntos Linearmente Separáveis

Um conjunto de treinamento S é linearmente separável se é possível separar os padrões das classes diferentes contidos no mesmo por pelo menos um hiperplano (Russel and Norvig, 1995).

Classificadores que separam os dados por meio de um hiperplano são denominados lineares, podendo ser definidos pela Equação 3.1.

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{3.1}$$

Onde $\mathbf{w} \cdot \mathbf{x}$ é o produto escalar entre os vetores $\mathbf{w}$ e $\mathbf{x}$, em que $\mathbf{w}$ é o vetor normal ao hiperplano e b é um termo "compensador". O par $(\mathbf{w}, b)$ é determinado durante o treinamento do classificador.

Esta equação divide o espaço de entradas em duas regiões: $\mathbf{w} \cdot \mathbf{x} + b > 0$ e $\mathbf{w} \cdot \mathbf{x} + b < 0$,

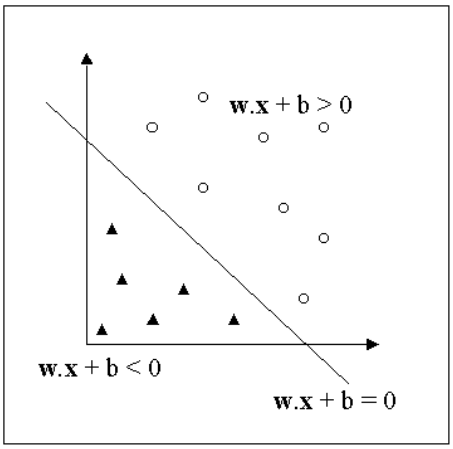

Figura 3.1: Conjunto de dados linearmente separável e um possível hiperplano separador.

levando à Equação 3.2 (Müller et al., 2001).

$$\begin{cases} y_i = +1 \; se \; \mathbf{w} \cdot \mathbf{x}_i + b > 0 \\ y_i = -1 \; se \; \mathbf{w} \cdot \mathbf{x}_i + b < 0 \end{cases} \tag{3.2}$$

Uma função sinal $g(\mathbf{x}) = \text{sgn}(f(\mathbf{x})) = \text{sgn}(\mathbf{w} \cdot \mathbf{x} + b)$, definida no Capítulo anterior, pode ser então aplicada sobre essa função, levando à classificação +1 se $f(\mathbf{x}) > 0$ e -1 se $f(\mathbf{x}) < 0$.

Logo, um conjunto de treinamento é linearmente separável se é possível determinar pelo menos um par $(\mathbf{w}, b)$ tal que função sinal $g(\mathbf{x}) = \text{sgn}(f(\mathbf{x}))$ consiga classificar corretamente todos os exemplos contidos neste grupo. Este fato pode ser observado na Figura 3.1.

3.2 SVMs Lineares

Na aplicação da TAE, deve-se escolher o classificador com o menor risco empírico possível e que também satisfaça a restrição de pertencer a uma família F com dimensão VC pequena, conforme discutido no Capítulo 2.

A primeira condição, no caso de conjuntos de treinamento linearmente separáveis, é satisfeita para pelo menos um par $(\mathbf{w}, b)$ definido pela Equação 3.1. Em geral, há diversas combinações de valores de $\mathbf{w}$ e b capazes de separar corretamente os dados deste tipo de conjunto.

Para satisfazer a segunda restrição, utiliza-se o resultado da TAE que relaciona o risco funcional de uma função à margem (ρ) de separação entre os dados de treinamento e o hiperplano separador. Uma definição formal do conceito de margem é apresentada a seguir (Smola et al., 1999b).

Seja f uma hipótese utilizada para classificação de entradas na forma $(\mathbf{x}_i, y_i)$, em que y_i representa a classe do padrão $\mathbf{x}_i$. Então a Equação 3.3 define a margem com a qual o padrão $\mathbf{x}_i$ é classificado. A margem γ de um classificador é dada então pela Equação 3.4.

$$\rho_f(\mathbf{x}_i, y_i) = yf(\mathbf{x}_i) \tag{3.3}$$

$$\rho = \min(y_i f\left(\mathbf{x}_i\right)) \tag{3.4}$$

Portanto, entre os classificadores que minimizam o risco empírico, deve-se escolher aquele que possui a maior margem $\rho^{'}$. O hiperplano que possui esta margem $\rho^{'}$ é denominado ótimo.

3.3 Determinação do Hiperplano Ótimo

Esta Seção apresenta os passos seguidos por uma SVM linear na determinação do hiperplano ótimo para conjuntos linearmente separáveis.

Como assumiu-se que o conjunto de treinamento é linearmente separável, pode-se reescalar $\mathbf{w}$ e b de forma que os pontos mais próximos do hiperplano separador satisfaçam $|\mathbf{w}\cdot\mathbf{x}+b=1|$ (Müller et al., 2001). Obtém-se com isto a representação canônica do hiperplano, adotada para facilitar as considerações subseqüentes realizadas na determinação do hiperplano ótimo.

A partir dessa consideração, a desigualdade 3.5 caracteriza os classificadores lineares que separam o conjunto de treinamento com uma margem positiva. Segundo este sistema, não há pontos entre $\mathbf{w}\cdot\mathbf{x}+b=0$ e $\mathbf{w}\cdot\mathbf{x}+b=\pm 1$, ou seja, supõe-se que a margem ρ é sempre maior que a distância entre os hiperplanos $\mathbf{w}\cdot\mathbf{x}+b=0$ e $|\mathbf{w}\cdot\mathbf{x}+b=1|$ (Campbell, 2000). Devido a essa suposição, as SVMs obtidas são usualmente denominadas SVMs com margens rígidas.

$$\begin{cases} \mathbf{w}\cdot\mathbf{x_i}+b \geq +1 \; se \; y_i = +1 \\ \mathbf{w}\cdot\mathbf{x_i}+b \leq -1 \; se \; y_i = -1 \\ \qquad i = 1, \; \ldots, \; n \end{cases} \tag{3.5}$$

Sejam $\mathbf{x}_1$ um ponto sobre $\mathbf{w}\cdot\mathbf{x}+b=-1$ e $\mathbf{x}_2$ um ponto sobre $\mathbf{w}\cdot\mathbf{x}+b=+1$ (Equação 3.6). Supõe-se também que $\mathbf{x}_1$ intercepta a reta perpendicular a $\mathbf{x}_2$, como indicado na Figura 3.2 (Hearst et al., 1998). Tem-se assim que:

$$\begin{cases} \mathbf{w}\cdot\mathbf{x}_1+b=-1 \\ \mathbf{w}\cdot\mathbf{x}_2+b=1 \end{cases} \tag{3.6}$$

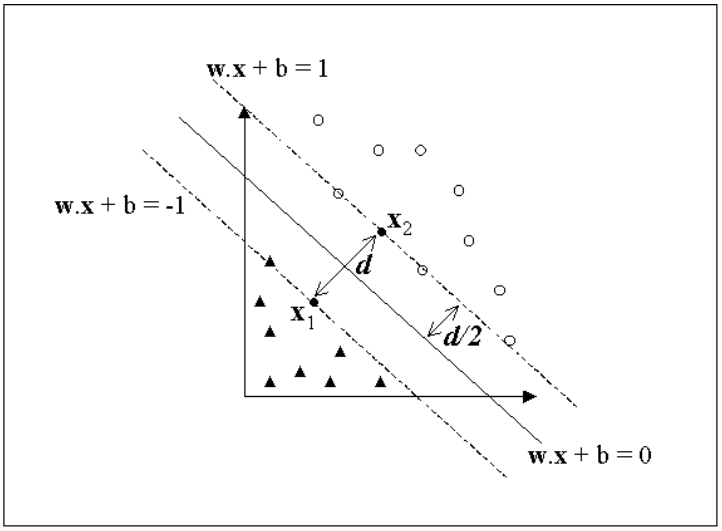

Figura 3.2: Ilustração da distância d entre os hiperplanos $\mathbf{w} \cdot \mathbf{x}_1 + b = -1$ e $\mathbf{w} \cdot \mathbf{x}_2 + b = +1$.

A partir do sistema 3.6, obtém-se a Equação 3.7.

$$\mathbf{w} \cdot (\mathbf{x}_2 - \mathbf{x}_1) = 2 \tag{3.7}$$

Como $\mathbf{w}$ e $\mathbf{x}_2 - \mathbf{x}_1$ são ortogonais ao hiperplano separador, esses vetores são paralelos entre si. Pode-se desta forma deduzir a Equação 3.8.

$$|\mathbf{w} \cdot (\mathbf{x}_2 - \mathbf{x}_1)| = \|\mathbf{w}\| \times \|\mathbf{x}_2 - \mathbf{x}_1\| \tag{3.8}$$

em que $\|\cdot\|$ representa a norma de um vetor. Substituindo 3.7 em 3.8, obtém-se a Equação 3.9.

$$\|\mathbf{x}_2 - \mathbf{x}_1\| = \frac{2}{\|\mathbf{w}\|} \tag{3.9}$$

A norma $\|\mathbf{x}_2 - \mathbf{x}_1\|$ mede a distância entre os hiperplanos $\mathbf{w} \cdot \mathbf{x} + b = -1$ e $\mathbf{w} \cdot \mathbf{x} + b = +1$. Logo, pela Equação 3.9 pode-se afirmar que a distância entre os mesmos é dada por $2/\|\mathbf{w}\|$. Da mesma forma, pode-se deduzir que a distância entre os hiperplanos $\mathbf{w} \cdot \mathbf{x} + b = 0$ e $\mathbf{w} \cdot \mathbf{x} + b = 1$ (ou $\mathbf{w} \cdot \mathbf{x} + b = -1$) é dada por $1/\|\mathbf{w}\|$. Como foi suposto que a margem é sempre maior que esta última distância, a minimização de $\|\mathbf{w}\|$ leva a uma maximização da margem.

Portanto, o hiperplano ótimo é definido para os valores de $\mathbf{w}$ e b que satisfazem as desigualdades 3.5 e para os quais a norma $\|\mathbf{w}\|$ é mínima. Este é um problema de otimização com restrições e pode ser reescrito como:

Minimizar: $\|\mathbf{w}\|^2$

Sob as restrições: $y_i(\mathbf{w} \cdot \mathbf{x_i} + b) - 1 \geq 0$, para $i = 1, \ldots, n$

Este é um problema clássico em otimização denominado *programação quadrática* (Hearst et al., 1998). O problema de AM é então reduzido a uma forma que pode ser analisada sob o ponto de vista de otimização de função quadrática, para o qual há uma ampla e estabelecida teoria existente. Na resolução do mesmo, introduz-se uma função Lagrangiana[1], definida em termos de $\mathbf{w}$ e b, apresentada em 3.10 (Campbell, 2000).

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{n} \alpha_i \left(y_i \left(\mathbf{w} \cdot \mathbf{x_i} + b\right) - 1\right) \tag{3.10}$$

em que os α_i são denominados *multiplicadores de Lagrange*. O problema passa a ser então a minimização de 3.10 em relação a $\mathbf{w}$ e b e a maximização dos α_i. Os pontos ótimos desta equação são obtidos por meio da resolução das igualdades apresentadas em 3.11 e 3.12.

$$\frac{\partial L}{\partial b} = 0 \tag{3.11}$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \tag{3.12}$$

As Equações 3.11 e 3.12 levam ao resultado representado por 3.13 e 3.14.

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3.13}$$

$$\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x_i} \tag{3.14}$$

Substituindo 3.13 e 3.14 em 3.10, obtém-se o seguinte problema de otimização (denominado dual):

$$\textbf{Maximizar: } \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x_i} \cdot \mathbf{x_j}$$

$$\textbf{Sujeito a: } \begin{cases} \alpha_i \geq 0,\ i = 1,\ \ldots,\ n \\ \sum_{i=1}^{n} \alpha_i y_i = 0 \end{cases}$$

Calculado o valor ótimo α^*, $\mathbf{w}^*$ é encontrado por 3.14[2]. Sendo determinada a direção

[1] O Apêndice A apresenta como um problema de otimização com restrições é solucionado através da introdução de uma função Lagrangiana.

[2] Os valores ótimos das variáveis serão indicados por *.

do hiperplano ótimo (ou seja, $\mathbf{w}^*$), b^* pode ser calculado por 3.15.

$$\begin{aligned} b^* &= -\tfrac{1}{2}\left[\max_{\{i|y_i=-1\}}(\mathbf{w}^*\cdot\mathbf{x_i}) + \min_{\{i|y_i=+1\}}(\mathbf{w}^*\cdot\mathbf{x_i})\right] \\ &= -\tfrac{1}{2}\left[\max_{\{i|y_i=-1\}}\left(\sum_{j=1}^{n} y_j\alpha_j^*\mathbf{x_i}\cdot\mathbf{x_j}\right) + \min_{\{i|y_i=+1\}}\left(\sum_{j=1}^{n} y_j\alpha_j^*\mathbf{x_i}\cdot\mathbf{x_j}\right)\right] \end{aligned} \tag{3.15}$$

Resumindo os procedimentos matemáticos realizados, o Algoritmo 3.1 apresenta o problema derivado e utilizado na determinação do hiperplano ótimo.

Algoritmo 3.1 Determinação do hiperplano ótimo para conjuntos linearmente separáveis (Vert, 2001).

1: Para cada conjunto de treinamento linearmente separável $S = \{(\mathbf{x}_1, y_1), \ \ldots, (\mathbf{x}_n, y_n)\}$
2: Seja $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ a solução do seguinte problema de otimização com restrições:
3: Maximizar: $\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x_i}\cdot\mathbf{x_j}$
4: Sob as restrições: $\begin{cases} \sum_{i=1}^{n} y_i\alpha_i = 0 \\ \alpha_i \geq 0,\ i = 1, \ \ldots, \ n \end{cases}$
5: O par $(\mathbf{w}^*, b^*)$ apresentado a seguir define o hiperplano ótimo.
6: $\mathbf{w}^* = \sum_{i=1}^{n} {\alpha_i}^* y_i \mathbf{x_i}$
7: $b^* = -1/2\left[\max_{\{i|y_i=-1\}}(\mathbf{w}^*\cdot\mathbf{x_i}) + \min_{\{i|y_i=+1\}}(\mathbf{w}^*\cdot\mathbf{x_i})\right]$

Segundo o Algoritmo 3.1, tenta-se encontrar os valores de α^* para utilizá-los na determinação de $(\mathbf{w}^*, b^*)$. É demonstrado que α_i^* assume valores positivos para exemplos do treinamento que estão a uma distância do hiperplano ótimo exatamente igual à margem. Para os outros padrões, o valor de α_i^* é nulo. Essa propriedade caracteriza uma esparsividade da solução (Cristianini and Shawe-Taylor, 2000).

Os exemplos para os quais $\alpha_i^* > 0$ são denominados *vetores suporte* (*Support Vectors* - SVs). A variável $\mathbf{w}^*$ pode então ser reescrita na forma indicada em 3.16.

$$\mathbf{w}^* = \sum_{\mathbf{x_i}\,\in\,SV} \alpha_i y_i \mathbf{x_i} \tag{3.16}$$

Como conseqüência, o hiperplano ótimo é determinado unicamente pelos SVs, que por este motivo são considerados os exemplos mais informativos do conjunto de treinamento.

Os padrões para os quais $\alpha_i = 0$ não participam na definição do hiperplano ótimo. Logo, se apenas o subconjunto dos dados de treinamento formado pelos SVs fossem empregados na indução, o classificador obtido seria o mesmo gerado com a utilização de todo o conjunto.

Fixados os valores dos parâmetros $\mathbf{w}^*$ e b^* por meio do treinamento de uma SVM, o classificador extraído é definido pela função 3.17.

$$g(\mathbf{x}) = \text{sgn}(f(\mathbf{x})) = \text{sgn}(\sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^*) = \begin{cases} +1 \ se \sum\limits_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^* > 0 \\ -1 \ se \sum\limits_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^* < 0 \end{cases} \quad (3.17)$$

A partir de 3.17, verifica-se que a classificação de um novo padrão $\mathbf{x}$ requer apenas a computação do produto interno entre $\mathbf{x}$ e cada SV.

3.4 Considerações Finais

Este Capítulo apresentou a forma mais simples de SVMs, também denominadas SVMs com margens rígidas, utilizada na geração de classificadores para conjuntos de treinamento linearmente separáveis.

Um conjunto linearmente separável é composto por exemplos que podem ser separados por pelo menos um hiperplano. As SVMs lineares buscam o hiperplano ótimo segundo a TAE apresentada no Capítulo 2, definido como aquele em que a margem de separação entre as classes presentes nos dados é maximizada.

O classificador apresentado, porém, não é eficiente quando confrontado com bases de dados mais gerais, podendo se orientar por ruídos presentes nos dados[3]. Este fato pode ser observado através das Equações 3.3 e 3.4. Um padrão $\mathbf{x}_i$ com rótulo incorreto na base de dados deverá ter margem negativa. Sendo sua classe y_i positiva, por exemplo, a previsão $f(\mathbf{x}_i)$ gerada pelo classificador deverá ser negativa, implicando em um valor negativo para a margem $\rho_f(\mathbf{x}_i, y_i)$ (Equação 3.3). Como a margem do classificador é dada pelo mínimo entre os $\rho_f(\mathbf{x}_i, y_i)$, esta também será negativa (Equação 3.4). Logo, o classificador se orientaria por eventuais erros contidos nos dados.

O Capítulo 4 estende os conceitos aqui apresentados para conjuntos de treinamento mais gerais, o que é realizado através de uma suavização das retrições impostas nas margens de separação entre os dados.

[3]Ruídos são definidos como dados imperfeitos (Baranauskas and Monard, 2000). Comumente, essas imperfeições são decorrentes de classificações incorretas de alguns padrões durante a geração dos conjuntos de treinamento para os algoritmos de aprendizado.

Capítulo 4

SVMs Lineares com Margens Suaves

Em situações reais, é difícil encontrar aplicações cujos conjuntos sejam linearmente separáveis. Isso se deve a diversos fatores, entre eles a presença de ruídos nos dados ou a própria natureza do problema, que pode ser não-linear.

Este Capítulo estende os conceitos introduzidos no Capítulo 3 para lidar com conjuntos de treinamento mais gerais, através do artifício de suavização de margens. Os classificadores gerados ainda serão lineares e será admitida, portanto, a ocorrência de alguns erros de classificação.

4.1 Adaptação das SVMs lineares com margens rígidas

Supôs-se anteriormente que era possível determinar $\mathbf{w}$ e b tal que houvesse uma região entre $\mathbf{w} \cdot \mathbf{x} + b = 0$ e $\mathbf{w} \cdot \mathbf{x} + b = \pm 1$ sem exemplos de treinamento. No caso de conjuntos não linearmente separáveis, como o apresentado na Figura 4.1, é mais difícil obedecer a esta restrição. Na tentatica de lidar com estes casos, introduz-se o conceito de variáveis de relaxamento ς (Figura 4.2), definidas pelas equações 4.1 e 4.2 (Vert, 2001). As variáveis de relaxamento suavisam as restrições impostas na determinação do hiperplano ótimo, sendo admitida com seu uso a ocorrência de alguns erros de classificação.

$$Para\, y_i = +1 \qquad \varsigma_i\left(\mathbf{w}, b\right) = \begin{cases} 0 & se\, \mathbf{w} \cdot \mathbf{x_i} + b \geq 1 \\ 1 - \mathbf{w} \cdot \mathbf{x_i} + b & se\, \mathbf{w} \cdot \mathbf{x_i} + b < 1 \end{cases} \tag{4.1}$$

$$Para\, y_i = -1 \qquad \varsigma_i\left(\mathbf{w}, b\right) = \begin{cases} 0 & se\, \mathbf{w} \cdot \mathbf{x_i} + b \leq -1 \\ 1 + \mathbf{w} \cdot \mathbf{x_i} + b & se\, \mathbf{w} \cdot \mathbf{x_i} + b > -1 \end{cases} \tag{4.2}$$

As variáveis ς_i medem onde se encontram os exemplos $(\mathbf{x}_i, y_i)$ em relação aos hiper-

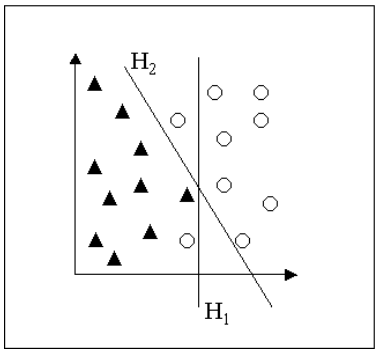

Figura 4.1: Exemplo de conjunto não linearmente separável e dois classificadores lineares H_1 e H_2 gerados sobre o mesmo.

planos $\mathbf{w} \cdot \mathbf{x} + b = \pm 1$. Se seu valor for 0, o exemplo está fora da região entre estes hiperplanos (é classificado corretamente) e se for positivo, mede a distância do padrão em relação aos mesmos. Outro ponto a ser observado se dá quando o dado é classificado errôneamente. Neste caso, ς_i assume valor maior que 1.

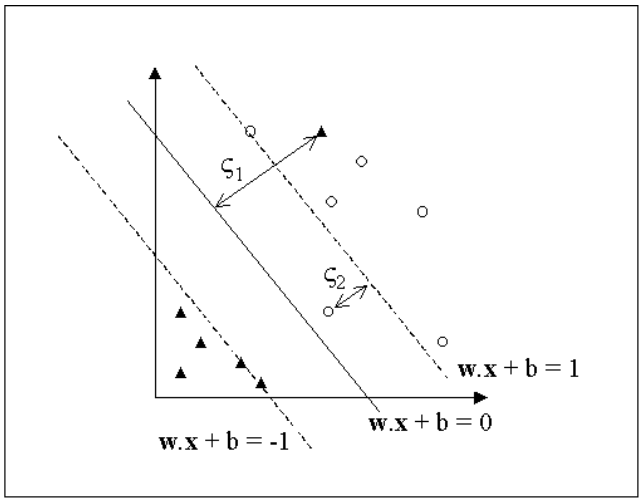

Figura 4.2: As variáveis de relaxamento ς_i.

A inclusão de variáveis de relaxamento é também denominada técnica de margens suaves (Cortes and Vapnik, 1995). Por meio desse artifício se evita que a SVM ajuste o hiperplano segundo ruídos nos dados.

Como observado para o caso de conjuntos linearmente separáveis, na determinação de um bom classificador linear procura-se obter o menor número possível de erros no treinamento e maximizar a margem de separação entre as classes (para haver uma boa generalização).

Para atender ao primeiro requisito no caso de conjuntos mais gerais, as variáveis de

relaxamento ς_i devem ter valor mínimo para todo o conjunto de treinamento. Já a segunda restrição é resolvida através da minimização de $\|\mathbf{w}\|$, de maneira similar ao realizado anteriormente.

Os valores a serem minimizados podem ser combinados através da Equação 4.3 (Campbell, 2000).

$$\epsilon(\mathbf{w}, b) = \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \varsigma_i(\mathbf{w}, b) \tag{4.3}$$

em que C é uma constante que impõe um peso diferente para o treinamento em relação à generalização e deve ser determinada empiricamente.

4.2 Determinação do Hiperplano Ótimo

A partir das considerações realizadas anteriormente, a determinação do hiperplano ótimo se resume a encontrar o par $(\mathbf{w}^*, b^*)$ que minimize a Equação 4.3.

Como as funções $\varsigma_i(\mathbf{w}, b)$ não são diferenciáveis em $\mathbf{w}$ e b, o problema é reformulado da seguinte maneira (Vert, 2001):

$$\textbf{Minimizar: } \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$$
$$\textbf{Sob as restrições: } \xi_i \geq \varsigma_i(\mathbf{w}, b),\ i = 1, \ldots, n$$

Utilizando as Equações 4.1 e 4.2, a restrição pode ser reescrita como (Smola et al., 1999b):

$$\begin{cases} \xi_i \geq 0 \\ y_i(\mathbf{w} \cdot \mathbf{x_i} + b) \geq 1 - \xi_i \end{cases} \tag{4.4}$$

Como no caso de conjuntos linearmente separáveis, chegou-se a um problema de otimização quadrática com restrições. A sua resolução envolve os mesmos passos matemáticos apresentados para o caso dos conjuntos linearmente separáveis (ou seja, introduz-se uma função Lagrangiana e otimiza-se um problema dual). Os resultados obtidos são semelhantes aos listados anteriormente, com exceção das seguintes modificações:

- A restrição $\alpha_i \geq 0$ é substituída por $0 \leq \alpha_i \leq C$, para $i = 1, \ldots, n$.
- A definição dos SVs é realizada segundo condições um pouco modificadas. Estas condições são descritas no próximo parágrafo.

Grande parte dos problemas de otimização são baseados em condições denominadas Karush-Kuhn-Tucker (KKT), necessárias para que o conjunto de soluções seja ótimo (Müller et al., 2001). Para o problema dual das SVMs com margens suaves, as condições

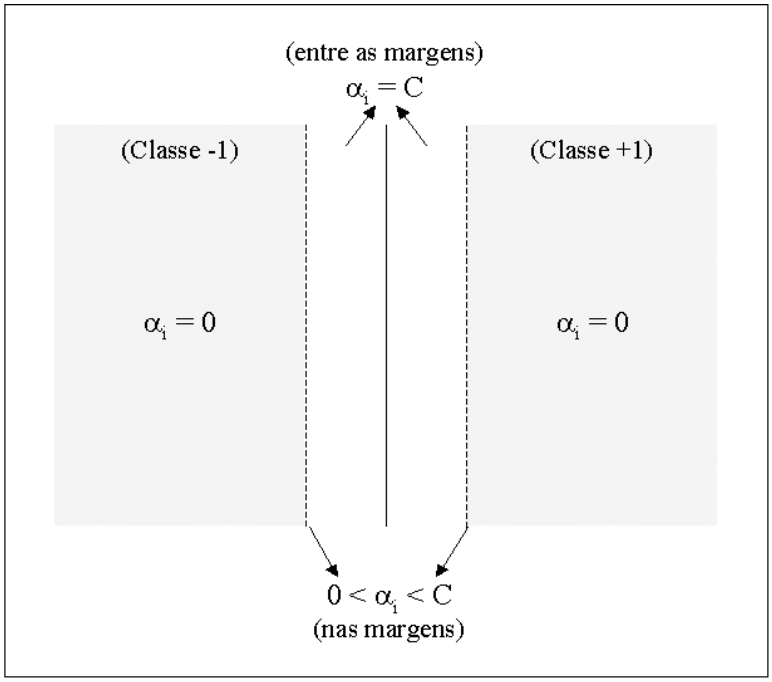

Figura 4.3: Relação dos valores dos multiplicadores de Lagrange α_i com as classificações geradas por uma SVM (Almeida et al., 2000). Padrões sobre a margem têm seus multiplicadores de Lagrange α_i com valor no intervalo $0 < \alpha_i < C$. Padrões que se encontram entre as margens têm $\alpha_i = C$. Os dois grupos descritos correspondem aos SVs. Para os demais padrões, $\alpha_i = 0$

são apresentadas abaixo, em que $y_i f(\mathbf{x_i})$ é a margem de classificação do exemplo i (Almeida et al., 2000).

(i) $\alpha_i = 0 \Rightarrow y_i f(\mathbf{x_i}) \geq 1 \, e \, \xi_i = 0$
(ii) $0 < \alpha_i < C \Rightarrow y_i f(\mathbf{x_i}) = 1 \, e \, \xi_i = 0$
(iii) $\alpha_i = C \Rightarrow y_i f(\mathbf{x_i}) < 1 \, e \, \xi_i \geq 0$

Os casos (ii) e (iii), em que os multiplicadores de Lagrange possuem valor positivo, correspondem aos SVs (a determinação do hiperplano ótimo pelas SVMs se dá unicamente em função desses padrões). Em (ii), tem-se a representação de um SV sobre a margem e em (iii), um SV entre as margens. Para os demais padrões, o valor do multiplicador de Lagrange associado é nulo (i). A Figura 4.3 apresenta uma representação dos casos descritos.

Os principais resultados rumo a determinação do hiperplano ótimo para conjuntos gerais são apresentados de forma explícita no Algoritmo 4.1.

Resolvido o problema apresentado no Algoritmo 4.1, a classificação de um novo padrão envolve apenas verificar o sinal da Equação 4.5, de forma similar à descrita no Capítulo

Algoritmo 4.1 Determinação do hiperplano ótimo para conjuntos de treinamento gerais (Vert, 2001).

1: Para cada conjunto de treinamento $S = \{(\mathbf{x}_1, y_1), \ \ldots, (\mathbf{x}_n, y_n)\}$
2: Seja $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ a solução do seguinte problema de otimização com restrições:
3: Maximizar: $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{x_i} \cdot \mathbf{x_j}$
4: Sob as restrições: $\begin{cases} \sum_{i=1}^{n} y_i \alpha_i = 0 \\ 0 \leq \alpha_i \leq C, \ i = 1, \ \ldots, \ n \end{cases}$
5: O par $(\mathbf{w}^*, b^*)$ apresentado a seguir define o hiperplano ótimo.
6: $\mathbf{w}^* = \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{x_i}$
7: $b^* = -1/2 \left[\max_{\{i | y_i = -1\}} (\mathbf{w}^* \cdot \mathbf{x_i}) + \min_{\{i | y_i = +1\}} (\mathbf{w}^* \cdot \mathbf{x_i}) \right]$

3.

$$g(\mathbf{x}) = \text{sgn}(f(\mathbf{x})) = \text{sgn}\left(\sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^* \right) = \begin{cases} +1 \ se \sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^* > 0 \\ -1 \ se \sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{x_i} \cdot \mathbf{x} + b^* < 0 \end{cases} \quad (4.5)$$

4.3 Considerações Finais

Neste Capítulo as SVMs lineares com margens rígidas apresentadas no Capítulo 3 foram adaptadas para lidar com conjuntos de treinamento não linearmente separáveis. Para isso, é admitida a ocorrência de alguns erros de classificação para o conjunto de treinamento, por meio da introdução de variáveis de relaxamento. Essas modificações fazem parte de um processo usualmente denominado "suavização de margens" (Cortes and Vapnik, 1995).

Apesar deste novo algoritmo se mostrar melhor que o anterior em situações práticas, seu uso ainda é limitado. Existem aplicações em que uma fronteira de decisão não linear é mais adequada para separação dos dados. O Capítulo 5 apresenta a forma como as SVMs lineares podem ser estendidas para lidar com estes casos.

Capítulo 5

SVMs Não Lineares

Os classificadores lineares descritos nos capítulos anteriores são bastante simples. Entretanto, sua utilização é limitada. Há muitos casos em que não é possível dividir satisfatoriamente os dados de treinamento por um hiperplano. Um exemplo é apresentado na Figura 5.1a, em que o uso de uma fronteira curva na separação das classes seria mais adequada. Este Capítulo descreve como as SVMs lineares podem ser generalizadas para lidar com tais situações.

Na realização desta tarefa, são definidas funções reais $\Phi_1, \ldots, \Phi_M$ no domínio do espaço de entrada. Por meio da utilização destas funções, mapea-se o conjunto de treinamento S (Equação 5.1) para um novo espaço, denominado *espaço de características*, da maneira apresentada em 5.2.

$$\begin{gathered} S = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\} \\ \text{em que } \mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{im}) \;\; e \; y_i \in \{-1, +1\} \end{gathered} \tag{5.1}$$

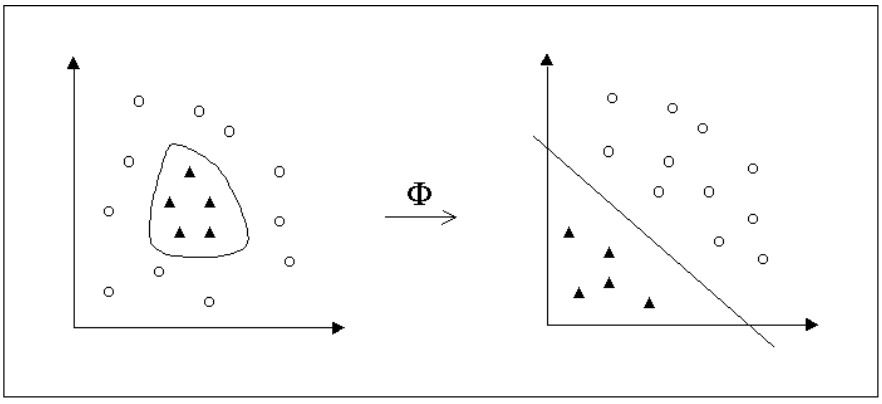

Figura 5.1: Transformação do conjunto de dados no espaço de entrada **(a)** para o espaço de características **(b)**.

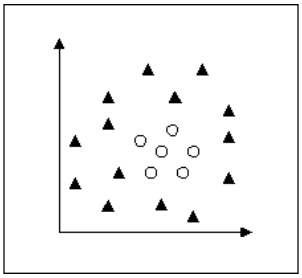

Figura 5.2: Exemplo de conjunto não linearmente separável.

$$\begin{aligned} \mathbf{x}_i \ (i = 1, \ldots, n) \quad &\mapsto \quad \mathbf{\Phi}\left(\mathbf{x}_i\right) = \left(\Phi_1\left(\mathbf{x}_i\right), \Phi_1\left(\mathbf{x}_i\right), \ldots, \Phi_M\left(\mathbf{x}_i\right)\right) \\ \Rightarrow \quad &\mathbf{\Phi}\left(S\right) = \left\{\left(\mathbf{\Phi}\left(\mathbf{x}_1\right), y_1\right), \left(\mathbf{\Phi}\left(\mathbf{x}_2\right), y_2\right), \ldots, \left(\mathbf{\Phi}\left(\mathbf{x}_n\right), y_n\right)\right\} \end{aligned} \tag{5.2}$$

Estas funções Φ_i podem ser não-lineares. Em particular, M pode ser muito maior que m, a dimensão do espaço de $\mathbf{x}$. Uma característica singular do espaço de características é que a escolha de uma função $\mathbf{\Phi}$ apropriada torna o conjunto de treinamento linearmente separável. As SVMs lineares anteriormente apresentadas podem então ser utilizadas sobre o conjunto de treinamento mapeado nesse espaço (Hearst et al., 1998). Este fato é ilustrado na Figura 5.1.

Considere, por exemplo, os dados representados na Figura 5.2 (Vert, 2001). Neste caso um limite circular é mais adequado para separação das classes. A Equação 5.3 define essa fronteira.

$$x_1^2 + x_2^2 < r^2 \tag{5.3}$$

Utilizando a transformação definida pela Equação 5.4 sobre a Equação 5.3, tem-se como resultado uma classificação linear em $\Re^3$. Esta é descrita por 5.5, em que $\mathbf{w} = (1, 0, 1)$ e $b = r^2$.

$$\mathbf{\Phi}\left(\mathbf{x}\right) = \left(x_1^2, x_1 x_2, x_2^2\right) \tag{5.4}$$

$$\mathbf{w} \cdot \mathbf{\Phi}\left(\mathbf{x}\right) + b = 0 \tag{5.5}$$

Portanto, mesmo que a fronteira entre as classes seja polinomial no espaço de entradas, este problema pode ser mapeado para outro de classificação linear no espaço de características.

Logo, os conceitos apresentados nos Capítulos anteriores podem ser estendidos para o caso não linear, através da definição de uma função de mapeamento $\mathbf{\Phi}$ adequada.

5.1 Trabalhando com SVMs lineares no espaço de características

Como os dados no espaço de características podem ser eficientemente separados por meio de fronteiras lineares, é possível utilizar o SVM linear apresentado no Algoritmo 4.1[1] para separar os padrões neste novo espaço. As adaptações realizadas são apresentadas explicitamente pelo Algoritmo 5.1.

Algoritmo 5.1 Determinação do hiperplano ótimo no espaço de características (Vert, 2001).

1: Para qualquer conjunto de treinamento $\mathbf{\Phi}(S) = \{(\mathbf{\Phi}(\mathbf{x}_1), y_1), \ldots, (\mathbf{\Phi}(\mathbf{x}_n), y_n)\}$
2: Seja $\alpha^* = (\alpha_1^*, \ldots, \alpha_n^*)$ a solução do seguinte problema de otimização com restrições:
3: Maximizar: $\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$
4: Sob as restrições: $\begin{cases} \sum_{i=1}^{n} y_i \alpha_i = 0 \\ 0 \leq \alpha_i \leq C, \ i = 1, \ \ldots, \ n \end{cases}$
5: O par $(\mathbf{w}^*, b^*)$ apresentado a seguir define o hiperplano ótimo.
6: $\mathbf{w}^* \leftarrow \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{\Phi}(\mathbf{x_i})$
7: $b^* \leftarrow -1/2 \left[\min_{\{i|y_i=+1\}} (\mathbf{w}^* \cdot \mathbf{\Phi}(\mathbf{x_i})) + \max_{\{i|y_i=-1\}} (\mathbf{w}^* \cdot \mathbf{\Phi}(\mathbf{x_i})) \right]$

O problema de classificação passa a ser a avaliação da função 5.6.

$$g(\mathbf{x}) = \text{sgn}(f(\mathbf{x})) = \text{sgn}\Big(\sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x}) + b^* \Big) = \begin{cases} +1 \ se \sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x}) + b^* > 0 \\ -1 \ se \sum_{\mathbf{x_i} \in SV} \alpha_i^* y_i \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x}) + b^* < 0 \end{cases} \tag{5.6}$$

Além disso, as considerações de Karish-Kuhn-Tucker apresentadas para o SVM linear no Capítulo 4 se mantêm para o caso não linear.

A partir das equações listadas no Algoritmo 5.1, percebe-se que a única informação necessária sobre o mapeamento $\mathbf{\Phi}$ é uma definição de como o produto interno $\mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$ pode ser realizado, para quaisquer $\mathbf{x_i}$ e $\mathbf{x_j}$ pertencentes ao espaço de entradas. Isto é obtido com a introdução do conceito de *Kernels*.

5.2 Funções Kernel

Um *Kernel* K é uma função que recebe dois pontos $\mathbf{x_i}$ e $\mathbf{x_j}$ do espaço de entradas e computa o produto escalar $\mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j})$ no espaço de características, como descrito em

[1] Utiliza-se o Algoritmo 4.1 (e não o 3.1) por este ser capaz de lidar com ruídos nos dados.

5.7 (Haykin, 1999).

$$K(\mathbf{x_i}, \mathbf{x_j}) = \mathbf{\Phi}(\mathbf{x_i}) \cdot \mathbf{\Phi}(\mathbf{x_j}) \tag{5.7}$$

Como exemplo tem-se que o Kernel do mapeamento representado pela Equação 5.4 é dado por $K(x_1, x_2) = (x_1 \cdot x_2)^2$ (Hearst et al., 1998). Pode-se observar que esta função é mais simples que a função $\mathbf{\Phi}$. Isso ocorre de maneira geral, pois em grande parte dos casos, $\mathbf{\Phi}$ assume formas bastante complexas. Por este motivo, é comum definir a função Kernel sem conhecer-se explicitamente o mapeamento $\mathbf{\Phi}$. A utilidade dos Kernels está, portanto, na simplicidade de cálculo e na capacidade de representar espaços muito abstratos.

As funções $\mathbf{\Phi}$ devem pertencer a um domínio em que seja possível o cálculo de produtos internos. Há um conjunto de funções com esta propriedade, que pertencem ao *espaço de Hilbert* (Wahba, 2000). Este conhecimento é utilizado na definição dos Kernels. Geralmente faz-se uso das condições estabelecidas pelo *Teorema de Mercer* nesta definição. Segundo este Teorema, os Kernels devem ser matrizes positivamente definidas, isto é, a matriz K, em que $K_{ij} = K(\mathbf{x_i}, \mathbf{x_j})$ para todo $i, j = 1, \ldots, n$, deve ter auto-valores maiores que 0 (Mercer, 1909). Essas funções são usualmente denominadas Kernels de Mercer (Smola et al., 1999b).

Alguns dos Kernels mais utilizados são os polinomiais, os Gaussianos ou RBF (*Radial-Basis Function*) e os Sigmoidais, apresentados na Tabela 5.1.

Tipo de Kernel	**Função $K(\mathbf{x_i}, \mathbf{x_j})$ correspondente**	**Comentários**
Polinomial	$\left(\mathbf{x_i}^T \cdot \mathbf{x_j} + 1\right)^p$	A potência p deve ser especificada pelo usuário
Gaussiano	$\exp\left(-\frac{1}{2\sigma^2}\left\|\mathbf{x_i} - \mathbf{x_j}\right\|^2\right)$	A amplitude σ^2 é especificada pelo usuário
Sigmoidal	$\tanh\left(\beta_0 \mathbf{x_i} \cdot \mathbf{x_j} + \beta_1\right)$	Utilizado somente para alguns valores de β_0 e β_1

Tabela 5.1: Sumário dos principais Kernels utilizados nas SVMs (Haykin, 1999).

Alguns pontos podem ser destacados em relação aos Kernels apresentados na Tabela 5.1 (Vert, 2001; Haykin, 1999; Burges, 1998):

- No caso de Kernels polinomiais, os mapeamentos $\mathbf{\Phi}$ também são funções polinomiais com complexidade crescente a medida que o expoente p aumenta.
- O Kernel Gaussiano corresponde a um espaço de características de dimensão infinita. Pode-se afirmar que quase todas formas de mapeamento podem ser implementadas por esta função em particular. Além disso, por meio da utilização desse tipo de

função, define-se Redes Neurais do tipo RBF (*Radial-Basis Function*), em que o número de funções base radiais e seus centros são determinados automaticamente pelo número de SVs e pelos valores de multiplicadores de Lagrange ($\alpha_\mathbf{i}$) associados aos mesmos, respectivamente[2].

- A utilização do Kernel Sigmoidal, por sua vez, permite explicitar uma Rede Neural Artificial do tipo *Perceptron* multicamadas. Neste caso, o número de neurônios da camada intermediária e os vetores de *bias* associados aos mesmos também são determinados pelo número de SVs e pelos valores dos ($\alpha_\mathbf{i}$) associados aos mesmos, respectivamente[3].

A obtenção de um classificador por meio do uso de SVMs envolve a escolha de uma função Kernel, além de parâmetros desta função e do algoritmo para determinação do hiperplano ótimo (como o valor da constante C, por exemplo). A escolha do Kernel e dos parâmetros considerados tem efeito no desempenho do classificador obtido (Müller et al., 2001), pois eles definem a fronteira de decisão induzida. Existem algumas técnicas para seleção do modelo, que provêm meios para determinação da função Kernel e parâmetros do algoritmo. Entre elas, tem-se a regra *span* de Chapelle and Vapnik (2000), a regra de Jaakola and Haussler (1999) e a regra proposta em Chapelle et al. (2002).

5.3 Considerações Finais

Como visto neste Capítulo, as SVMs lineares podem ser generalizadas de forma a realizar classificações não lineares. Para isto, utiliza-se o artifício de levar os dados de treinamento para um espaço de maior dimensão. Os dados neste espaço se tornam linearmente separáveis. Pode-se então aplicar as SVMs lineares sobre os dados mapeados neste espaço, determinando um hiperplano de margem maximizada.

A SVM não linear incorpora este conceito por meio da utilização de funções *Kernel*, que permitem o acesso a espaços complexos (em alguns casos infinitos) de forma simplificada.

Até o momento, foram apresentadas aplicações de SVMs para problemas de classificação binária. O próximo Capítulo apresenta alguns meios propostos para possibilitar o uso de SVMs em problemas com mais de duas classes.

[2]O número de funções base radiais e seus centros são parâmetros de Redes Neurais do tipo RBF determinados normalmente de forma empírica. Uma conceituação mais detalhada sobre este tipo de rede pode ser consultada em (Haykin, 1999; Braga et al., 2000).

[3]Em uma Rede Neural do tipo Perceptron Multicamadas o número de neurônios na camada intermediária e os valores de *bias* são normalmente obtidos de forma empírica. A conceituação deste tipo de rede pode ser consultada em (Haykin, 1999; Braga et al., 2000).

Capítulo 6

SVMs para Várias Classes

Como pôde ser observado nos Capítulos anteriores, as SVMs são originalmente utilizadas para classificação dos dados em duas classes distintas. Estas podem ser denominadas positivas e negativas, respectivamente. Contudo, muitas aplicações envolvem o agrupamento em mais de duas classes. Este fato não inviabiliza o uso das SVMs. Diversas técnicas são propostas para estendê-las a problemas desse tipo (*multiclasses*).

Em um problema multiclasses, o conjunto de treinamento é composto por pares $(\mathbf{x_i}, y_i)$, tal que $y_i \in 1, \ldots, k$. Este Capítulo aborda soluções para problemas em que $k > 2$.

De fato, qualquer método para gerar classificadores multiclasses a partir de preditores binários pode utilizar as SVMs como base (Mayoraz and Alpaydm, 1998). As Seções 6.1 e 6.2 apresentam duas abordagens usuais para realização dessa tarefa, denominadas, respectivamente, decomposição "um-contra-todos" e "todos-contra-todos".

6.1 Decomposição "um-contra-todos"

Para a solução de um problema multiclasses a partir de SVMs, uma abordagem usual consiste na geração de k SVMs, onde k é o número de classes (Weston and Watkins, 1998). Na criação de cada uma dessas máquinas, uma classe é fixada como positiva e as restantes como negativas. Esta metodologia é denominada decomposição "um-contra-todos" (1-c-t) e é independente do algoritmo de aprendizado utilizado no treinamento dos classificadores (Mayoraz and Alpaydm, 1998). Na predição da classe de um padrão $\mathbf{x}$, basta escolher a saída com valor máximo entre as k SVMs, conforme apresentado na Equação 6.1.

$$f(\mathbf{x}) = \arg\max_{1 \leq i \leq k} (\mathbf{w_i} \cdot \mathbf{\Phi}(\mathbf{x}) + b_i) \tag{6.1}$$

O método 1-c-t tem a desvantagem de não ser possível prever limites no erro de generalização através de seu uso. Além disso, seu tempo de treinamento usualmente é

longo.

6.2 Decomposição "todos-contra-todos"

Outra abordagem para solução de problemas multiclasses a partir de classificadores binários envolve a construção de $k(k-1)/2$ SVMs, separando cada classe de outra, método denominado "todos-contra-todos" (t-c-t). Para unir estes classificadores, Friedman (1996) propôs o uso de um esquema de votação por maioria, em que cada um dos classificadores fornece uma classe como resultado. A solução final é dada pela classe que recebeu mais indicações. Esta metodologia, porém, também não provê limites no erro de generalização. Além disso, o tamanho dos classificadores gerados é em geral grande e a avaliação de seu resultado pode ser lenta.

Na resolução desses problemas, Platt et al. (2000) sugerem a utilização de um grafo direcionado acíclico[1] (DAG), de forma que o problema de agrupamento em várias classes é decomposto em diversas classificações binárias em cada nó do grafo. Este processo é ilustrado na Figura 6.1.

A Figura 6.1 apresenta um exemplo com quatro classes. O processamento feito pelo DAG é equivalente a operação de uma lista. Inicialmente, a lista é composta por todas as classes. Em cada nó é gerada uma SVM que separa os elementos da primeira e última classe da lista. Escolhida uma das classes, a outra é eliminada da lista e o processamento continua da mesma forma, até que um nó folha seja atingido. Portanto, a classificação de um exemplo de um problema de k classes requer a avaliação de $k-1$ nós, ou seja, $k-1$ SVMs. Com isso o tempo de geração de resultados é reduzido.

Platt et al. (2000) demonstraram que essa técnica tem erro de generalização limitado pela margem máxima obtida em cada nó. Como as SVMs utilizam o princípio de maximização de margens, seu uso como indutor base é bastante adequado.

6.3 Considerações Finais

Embora as SVMs sejam originalmente produzidas para geração de classificadores binários, alguns artifícios permitem que estas sejam aplicadas a problemas multiclasses, em que o número de classes é maior que dois.

Duas abordagens usuais para tal são a decomposição "um-contra-todos" e "todos-contra-todos". Sendo k o número de classes, no primeiro caso produz-se k classificadores, cada um separando uma classe i das $k-1$ restantes. A classe de um novo padrão é

[1] Um grafo direcionado acíclico é um grafo sem ciclos, cujas arestas possuem uma orientação.

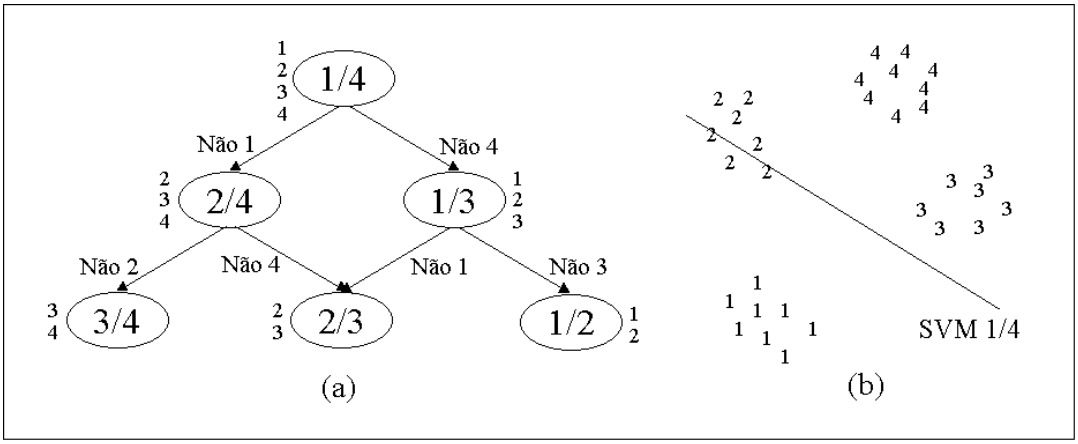

Figura 6.1: **(a)** Exemplo de grafo direcionado utilizado para classificar quatro classes a partir de SVMs binárias, as quais estão representados pelos nós da árvore (Platt et al., 2000). **(b)** Diagrama do espaço de características do problema 1/4.

dada pelo índice da SVM que produz a maior saída. No caso da decomposição "todos-contra-todos", são produzidos classificadores para separação de cada classe i de outra j, em que $i, j = 1, \ldots, k$ e $i \neq j$. Neste caso, a saída de um padrão é dada por um voto de maioria entre as SVMs. Os classificadores gerados por decomposição todos-contra-todos também podem ser unidos por meio de um grafo direcionado acíclico (DAGSVM) (Platt et al., 2000). Cada uma dessas alternativas tem vantagens e desvantagens, as quais foram brevemente discutidas neste Capítulo. Uma comparação mais detalhada dos métodos decomposicionais descritos neste Capítulo é apresentada em (Hsu and Lin, 2002).

Capítulo 7

Treinamento e Implementações de SVMs

O aspecto central na implementação das SVMs é a necessidade de resolver um problema de programação quadrática. Existem vários pacotes matemáticos que realizam esta tarefa. Exemplos incluem o MINOS (Murtagh and Saunders, 1993) e o LOQOS (Vanderbei, 1994). Porém, esses pacotes não são adequados a problemas com grandes quantidades de dados, pois requerem o armazenamento em memória da matriz Kernel, a qual é quadrática na quantidade de dados.

Há diversos trabalhos que visam simplificar este processo, os quais propõem novos algoritmos para o treinamento de SVMs. Grande parte das técnicas propostas derivam soluções explorando a esparsividade do problema (isto é, o fato de que grande parte dos multiplicadores de Lagrange α_i possuem valor nulo) e a convexidade da função a ser otimizada (Cristianini and Shawe-Taylor, 2000). A Seção 7.1 descreve brevemente alguns desses trabalhos.

Algumas implementações de SVMs disponíveis para uso científico são listadas na Seção 7.2.

7.1 Abordagens para o treinamento das SVMS

A alternativa mais direta para solucionar o problema de otimização das SVMs é utilizar uma estratégia gradiente ascendente (Cristianini and Shawe-Taylor, 2000).

Frieβ et al. (1998) apresentaram um método para o treinamento de SVMs que explora o uso deste tipo de estratégia, técnica denominada *Kernel Adatron*. Uma variante desse algoritmo, denominada *Kernel Adatron com bias e margens suaves* (KAbs), lida melhor com o problema de ruídos nos dados por meio da utilização do conceito de *bias* e margens

suaves (Campbell and Cristianini, 1998). O algoritmo KAbs combina a rapidez e simplicidade de uma Rede Neural com o poder preditivo das SVMs. Outra vantagem desse método está na facilidade de sua implementação e na possibilidade de um ajuste iterativo de alguns parâmetros dos Kernels, tal como a variável σ no caso de funções Gausianas.

Várias das outras alternativas para simplificar o processo de treinamento das SVMs compreendem métodos de decomposição do problema de otimização em subproblemas menores. Um exemplo é a técnica de *Chunking* (Vapnik, 1982). Este algoritmo explora a esparsividade do vetor de soluções α^*, ou seja, o fato de que o valor da função objetivo não muda caso colunas e linhas da matriz Kernel K que correspondem a valores α_i nulos sejam removidas. A cada iteração do algoritmo, somente os padrões de treinamento que são SVs (com $\alpha_i \neq 0$) e pontos que violam as condições de Karush-Kuhn-Tucker (KKT) são mantidos. As condições de KKT são condições necessárias e suficientes para garantir a convergência do problema de otimização quadrática. No caso das SVMs, essas condições são representadas pelo sistema 7.1[1]. No último passo do algoritmo de *Chunking*, restam somente os SVs. Esta técnica ainda é limitada pelo tamanho do problema, pois o número de SVs pode ser grande. Além disso, um pacote numérico ainda é necessário para solução dos subproblemas de otimização.

$$\begin{gathered} 0 \leq \alpha_i \leq C \\ y_i f(\mathbf{x_i}) \begin{cases} \geq 1 \text{ para pontos com } \alpha_i = 0 \\ = 1 \text{ para pontos com } 0 < \alpha_i < C \\ \leq 1 \text{ para pontos com } \alpha_i = C \end{cases} \end{gathered} \tag{7.1}$$

Osuna et al. (1997) propuseram outra técnica decomposicional que limita o tamanho dos subproblemas de otimização a serem solucionados a cada passo do algoritmo. Para isto, a técnica sugerida explora o fato de que uma seqüência de subproblemas de otimização quadrática que contenham no mínimo um exemplo violando as condições de KKT convergem para a solução ótima. Esta abordagem é semelhante a do *Chunking*, porém o tamanho dos subproblemas atacados é mantido fixo. A cada passo do algoritmo, o mesmo número de exemplos é adicionado e removido do problema. A heurística comumente utilizada neste processo consiste em adicionar os padrões que violam as condições de KKT e remover os exemplos para os quais α_i tem valor nulo ou igual a C. Com isto, permite-se lidar com problemas de maior tamanho. Na prática, porém, este algoritmo se mostra lento. Além disso, na resolução de cada subproblema de otimização, ainda é necessária a utilização de um programa de otimização. Existem trabalhos que utilizam heurísticas mais sofisticadas para escolha dos padrões a serem removidos e adicionados,

[1] As condições de KKT da Equação 7.1 são as mesmas utlizadas na determinação dos SVs para SVMs com margens suaves, apresentadas no Capítulo 4.

além de utilizarem métodos de *caching* (Müller et al., 2001). Com isto, tornam-se mais rápidos e adequados a problemas de maior escala. Um exemplo é a ferramenta SVMlight (Joachims, 1998), descrita na próxima Seção.

O caso mais extremo de decomposição do problema é realizado pelo algoritmo *Sequential Minimal Optimization* (SMO), de Platt (1999). A cada iteração do SMO, somente dois padrões são examinados, ou seja, soluciona-se um problema de otimização quadrática de tamanho 2 a cada ciclo do algoritmo. Este é o menor tamanho de problema de otimização possível, mantendo a restrição $\sum_{i=1}^{n} \alpha_i y_i = 0$. A cada ciclo, escolhidos os valores α_1 e α_2 a serem otimizados (lembrando que cada α_i corresponde diretamente a um padrão $\mathbf{x_i}$), os α_i remanescentes são mantidos constantes, e α_1 e α_2 são atualizados de forma que a obedecer a igualdade representada pela Equação 7.2 (Cristianini and Shawe-Taylor, 2000).

$$\alpha_1 y_1 + \alpha_2 y_2 = const. = \alpha_1^{antigo} y_1 + \alpha_2^{antigo} y_2 \tag{7.2}$$

Deve-se lembrar que a restrição $0 \leq \alpha_1, \alpha_2 \leq C$ deve ser obedecida. Os novos valores das variáveis α_1 e α_2 podem ser determinados analiticamente, sem a necessidade do uso de programas de otimização. Na escolha dos exemplos a serem analisados a cada ciclo, heurísticas são utilizadas. O padrão correspondente a α_1 é escolhido entre aqueles que violam as condições de KKT. O segundo padrão (α_2) é escolhido de forma que a atualização no par (α_1, α_2) acarrete um maior crescimento no valor da função objetivo. A implementação desta técnica é simples[2]. Além disso, este algoritmo geralmente é rápido.

A seguir uma lista, acompanhada de uma breve descrição, de algumas implementações de SVMs disponíveis é apresentada.

7.2 Implementações Disponíveis

Há uma grande variedade de implementações de SVMs disponíveis para uso não-comercial. Vários algoritmos podem ser acessados via (Smola and Schölkopf, 2000). Algumas destas ferramentas são brevemente descritas a seguir:

- **SMO:** uma implementação em C++ do algoritmo SMO para classificação foi desenvolvida por Ge (2001). Essa ferramenta deve ser utilizada em plataforma Unix/Linux. Como Kernel, estão disponíveis as funções Linear e Gaussiana. Esta implementação lida com problemas binários e multiclasses.

- **SVMlight:** implementação de Joachims (1998). Pode ser utilizada em problemas de classificação, binários ou multiclasses, e de regressão. Permite a utilização

[2]Um pseudocódigo do SMO pode ser consultado em (Cristianini and Shawe-Taylor, 2000).

de vários tipos de função Kernel: Linear, Polinomial, Gaussiano, Sigmoidal e de funções definidas pelo usuário. Compilada para plataformas Windows NT, Unix, Linux e Powermac. Usa ferramenta de programação quadrática para resolução dos problemas de otimização, distribuída juntamente com o software.

- **mySVM:** ferramenta baseada no SVMlight (Sewell, 2001). Utilizada em reconhecimento de padrões (classificação), sobre problemas binários ou multiclasses, e regressão. Pode ser utilizada em plataformas Windows, Unix e Linux. Possibilita a utilização de diversas funções Kernel, entre as quais: Linear, Polinomial, Gaussiana, RBF Anova, Sigmoidal e definidas pelo usuário, além permitir o uso de somas e produtos de Kernels.

- **LIBSVM:** biblioteca de funções para SVMs, com interface gráfica, projetada por Chang and Lin (2003). Possui fontes em C++ e Java. Também possui interface para Matlab. As implementações são baseadas nas ferramentas SMO e SVMlight. Permite seleção de modelo por *cross-validation* e atribuir pesos aos dados para lidar com distribuições de dados desbalanceadas. Pode ser utilizado em problemas de classifição multiclasses e regressão.

- **SVM *toolbox* para Matlab:** *toolbox* para Matlab (Schwaighofer, 2002) baseado no SVMlight. Pode ser utilizado em problemas de classificação binária ou multiclasses. Pode ser facilmente modificado e requer a utilização de um pacote de otimização, que pode ser o fornecido pelo Matlab.

- **SVMTorch:** adaptação do SVMlight, especial para lidar com problemas de larga escala (Collobert and Bengio, 2001). Pode ser utilizado em problemas de classificação (binária e multiclasses) e regressão. Pode ser utilizado em plataformas Unix, Linux e Windows. Incorpora Kernels Linear, Polinomial, Gaussiano, Sigmoidal e definidos pelo usuário.

Existem alguns estudos para comparação do desempenho de alguns dos algoritmos citados (Carvalho, 2002; Carrieras and Pérez, 2001; Chang et al., 2000). A adequabilidade de cada um, porém, depende do problema a ser investigado.

7.3 Considerações Finais

Este Capítulo listou alguns trabalhos que visam facilitar o treinamento das SVMs. Em geral esse processo é complexo, já que envolve a resolução de um problema de otimização quadrática. Embora existam diversos pacotes de otimização que solucionam este tipo de

problema, estes são inadequados para as aplicações em AM, que em geral caracterizam-se pela necessidade de lidar com um grande volume de dados.

Diversas técnicas foram então propostas para adaptar o problema de otimização das SVMs a aplicações de larga escala. Parte destes utilizam estratégias decomposicionais, em que subproblemas menores são otimizados a cada passo do algoritmo (ex.: (Vapnik, 1982; Osuna et al., 1997; Joachims, 1998)).

Algumas implementações de SVMs disponíveis também foram apresentadas.

A seguir, o Capítulo 8 apresenta o uso das SVMs em algumas aplicações práticas.

Capítulo 8

Aplicações

As SVMs vêm sendo aplicadas com sucesso a diversos problemas computacionais (Hearst et al., 1998; Smola et al., 1999b; Müller et al., 2001).

Existem diversos *benchmarks* para avaliação dos resultados obtidos por técnicas de AM. Em (Müller et al., 2001) um estudo comparativo entre diversas técnicas de AM, entre elas as SVMs, Redes Neurais Artificiais do tipo RBF e o AdaBoost[1] (AB), é apresentado. A Tabela 8.1 ilustra a taxa de erro observada no teste de classificadores gerados pelas técnicas citadas, sobre conjuntos de dados do repositório IDA (IDA, 2002). Na geração dos classificadores, os dados foram divididos segundo o método *10-fold cross validation*. Segundo este método, o conjunto de dados é dividido em 10 partições de tamanho aproximadamente igual. Nove destas partições são utilizadas no treinamento do classificador, enquanto a restante é utilizada em seu teste. Tem-se então um total de dez pares de conjuntos para treinamento e teste. O erro total do classificador é dado pela média dos erros observados em cada partição.

	Taxa de Erro de Classificação		
Base	SVM	RBF	AB
Cancer seio	$\mathbf{26.0 \pm 0.47}$	27.6 ± 0.47	30.4 ± 0.47
Diabetes	$\mathbf{23.5 \pm 0.17}$	24.3 ± 0.19	26.5 ± 0.23
Coração	$\mathbf{16.0 \pm 0.33}$	17.6 ± 0.33	20.3 ± 0.34
Splice	10.9 ± 0.07	$\mathbf{10.0 \pm 0.10}$	10.1 ± 0.05
Tireóide	4.8 ± 0.22	4.5 ± 0.21	$\mathbf{4.4 \pm 0.22}$

Tabela 8.1: Comparação entre os erros das técnicas SVM, RNA RBF e AB para diferentes bases de dados (Müller et al., 2001). Os melhores resultados são apresentados em negrito.

Através da Tabela 8.1 pode-se verificar a eficácia das SVMs na solução de diferentes

[1]O AdaBoost é uma técnica utilizada na geração de comitês de classificadores. Um comitê consiste de um conjunto de preditores cujas saídas são combinadas na previsão da classe de novas instâncias (Baranauskas and Monard, 2000). Na Tabela 8.1, RNAs foram utilizadas como preditores base.

tipos de problemas. Os resultados das SVMs são superiores ou similares aos das outras técnicas apresentadas.

As próximas Seções descrevem aplicações das SVMs em três áreas distintas: Visão Computacional, Reconhecimento de Dígitos e Bioinformática. É focada, porém, a descrição de trabalhos em Bioinformática, área em que o uso de SVMs tem se mostrado bastante promissor.

8.1 Visão Computacional

A área de pesquisa de Visão Computacional investiga alternativas para incorporar as características dos sistema de visão natural em sistemas computacionais. Busca-se extrair informações a partir das imagens, utilizando para isso recursos de Inteligência Artificial, processamento de imagens e reconhecimento de padrões (Dowling, 1996).

Uma aplicação bastante pesquisada é a identificação de faces humanas em imagens. Fernandez and Viennet (1999) aplicaram SVMs com Kernel do tipo RBF na realização desta tarefa. Nos experimentos realizados foram utilizadas 10 imagens de 40 pessoas. Foram gerados classificadores para as imagens originais e para os mesmos dados submetidos a dois conjuntos de pré-processamentos distintos. O primeiro deles era composto por reescalas e da técnica de Análise de Componentes Principais (PCA) (Diamantaras and Kung, 1996). O segundo, por translações e efeitos de zoom. Neste último caso, um subconjunto dos dados não foi submetido aos processamentos citados. Os erros obtidos durante o teste foram de 5.5%, 3.7% e 1.5%, respectivamente. Na comparação com outras técnicas de AM o melhor resultado foi de 2.7% para o erro, obtido por uma RNA.

Em Hearst et al. (1998) também é descrita uma aplicação de SVMs ao problema de detecção de faces em imagens. Para compensar variações nas imagens, as seguintes técnicas de pré-processamento foram aplicadas: máscara para diminuição da dimensão das imagens, correção de gradiente para redução de sombras e efeitos de luminosidade e equalização para lidar com diferenças de brilho e de curvatura de câmeras. Os resultados foram comparados aos de Sung and Poggio (1994), que utilizaram clusterização e RNAs para a mesma tarefa considerada. Os resultados descritos em Hearst et al. (1998) se mostraram similares e, em alguns casos, superiores.

8.2 Reconhecimento de Dígitos

Esta aplicação consiste em reconhecer um número a partir de dígitos manuscritos. Diversos trabalhos deste tipo foram conduzidos utilizando a base do Serviço Postal Americano (USPS) (Schapire et al., 1997; DeCoste and Schölkopf, 2001; LeCun et al., 1995).

Este repositório contém 9298 dígitos manuscritos na forma de matrizes de dimensão 16x16, cujas entradas possuem valores entre -1 e 1.

Em (Campbell, 2000), uma RNA do tipo RBF e uma SVM com Kernel RBF foram comparadas na resolução deste problema, utilizando a base de dados USPS para treinamento e teste dos classificadores gerados. Os resultados alcançados podem ser visualizados na Tabela 8.2. Verifica-se que a SVM mostrou um melhor desempenho no reconhecimento de todos os dígitos.

Dígito	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
RBF	20	16	43	38	46	31	15	18	37	26
SVM	16	8	25	19	29	23	14	12	25	16

Tabela 8.2: Número de erros obtidos na classificação de dígitos manuscritos pertencentes a um conjunto de teste com 2007 padrões (Campbell, 2000).

8.3 Bioinformática

Por Bioinformática entende-se a aplicação de técnicas computacionais, envolvendo desde o armazenamento de informações à análise das mesmas, no gerenciamento de informações biológicas (Baldi and Brunak, 1998). Os dados neste domínio em geral são volumosos e complexos (possuem grande número de atributos, ou seja, grande dimensão), o que torna o uso das SVMs adequadas.

Seguem algumas das principais aplicações de interesse em Bioinformática e como as SVMs forma empregadas em cada uma delas.

8.3.1 Identificação de genes

A tarefa de identificação de genes envolve o reconhecimento e a localização de cada gene em seqüências de DNA. Uma abordagem comumente utilizada na realização desta tarefa envolve determinar sinais que podem ser identificados nas seqüências e que indicam situações especialmente adequadas à localização de genes (Cravem and Shavlik, 1994).

Um exemplo de sinal é o sítio de início de tradução (SIT) em seqüências de mRNA (Ácido Ribonucleico mensageiro[2]). Zien et al. (2000) reportam o uso de SVMs no reconhecimento desses sítios. O desempenho das SVMs é comparado ao das RNAs. Resultados melhores são observados através da aplicação das SVMs. Outro estudo conduzido pelos mesmos autores envolveu a incorporação de informações específicas do domínio biológico

[2]O mRNA é gerado na Expressão Gênica, processo biológico estudado na Bioinformática.

(conhecimento *a priori*) às funções Kernel. A realização dessas modificações se mostrou bastante simples e os resultados alcançados pelas SVMs foram melhores que os obtidos com o uso de funções Kernel tradicionais. A utilização de informação a priori no desenvolvimento de funções Kernel é uma área de pesquisa bastante investigada atualmente, permitindo a obtenção de Kernels mais adequados às aplicações investigadas (Schölkopf et al., 1998).

Em organismos eucariotos[3], outros tipos de sinais são as junções de processamento. Essas junções delimitam regiões do gene que são traduzidos em proteínas, denominadas *exons*, e regiões que não são codificadas, os *introns*. Em (Lorena et al., 2002a) SVMs, RNAs e Árvores de Decisão (ADs) são aplicadas na identificação desses sítios. Melhores resultados são observados com a utilização das SVMs. Em (Lorena et al., 2002b) resultados semelhantes são obtidos na comparação entre SVMs e ADs.

8.3.2 Estrutura de proteínas

As funções e propriedades de uma proteína são determinadas por sua estrutura tridimensional (Casley, 1992). Esta estrutura, por sua vez, é definida pela seqüência de genes que codificam a proteína em questão. Uma área de pesquisa que tem despertado grande interesse é a análise dos dados genéticos seqüenciados com o fim de determinar a estrutura das proteínas codificadas. Estudos semelhantes são conduzidos para determinar a estrutura tridimensional de moléculas de RNA.

Uma das tarefas envolvidas na previsão da estrutura de proteínas é o reconhecimento de tipos de "dobras" que estas possuem. Métodos para comparação de seqüências como o *pairwise* são comumente utilizados nessa tarefa, buscando seqüências similares e atribuindo o tipo de dobra destas à proteína sendo consultada. Porém, existem bases com menos de 35% de semelhança entre as proteínas em nível seqüencial. Para estes casos, o uso de algoritmos de AM tem se mostrado mais adequado.

Ding and Dubchak (2001) empregaram SVMs e RNAs na solução desse problema. As RNAs também foram utilizadas como classificadores binários, a exemplo das SVMs. Embora essas estruturas possam ser utilizadas diretamente na classificação envolvendo mais de duas classes, é argumentado que as RNAs são mais precisas e eficientes quando lidam com problemas de classificação binária. Na geração de classificadores multiclasses a partir de dicotomias, foram utilizados os métodos 1-c-t, 1-c-t único (versão do 1-c-t adaptado para lidar com o problemas de falsos positivos[4]) e t-c-t. Devido aos longos períodos de treinamento requeridos pelas RNAs, a metodologia t-c-t não foi empregada,

[3]Organismos que possuem o material genético delimitado em um núcleo.

[4]Falsos positivos são padrões classificados como pertencentes à classe positiva, sendo sua classificação correta a negativa.

pois tornou-se proibitiva. É reportado que as SVMs obtiveram melhores resultados que as RNAs em todos os casos. Entre as SVMs, aquela em que se usou t-c-t foi a mais precisa.

8.3.3 Análise de Expressão Gênica

O objetivo neste caso é analisar o nível de expressão de diferentes genes (a "quantidade" de proteínas que este produz). Com isto pode-se, por exemplo, comparar mudanças na expressão de alguns genes frente a alguma doença, possibilitando assim identificar alvos para futuros medicamentos e/ou terapias gênicas (Furey et al., 2001).

Muitos trabalhos utilizam SVMs na análise de dados provenientes de experimentos de *DNA microarray*, técnica de laboratório utilizada para medir o nível de expressão de genes. Esses dados caracterizam-se por possuir uma grande dimensionalidade (possuem medidas de milhares de genes). As SVMs, portanto, são apropriadas a essa tarefa.

A medida do nível de expressão dos genes em geral é útil no diagnóstico de moléstias, por meio da identificação de genes relacionados a uma determinada doença. Furey et al. (2001) aplicaram SVMs sobre dados de *DNA microarray* para classificação de tecidos cancerosos e, por meio de adaptações no algoritmo original, determinaram exemplos do conjunto de treinamento que continham erros em sua classificação, que é usualmente feita a priori por um Biólogo. Os resultados observados se mostraram comparáveis a outros métodos, tais como as RNAs (Tamayo et al., 1996).

Em um experimento semelhante, Ben-Hur et al. (2000) aplicaram SVMs, AdaBoost e clusterização sobre dados de *microarray* para classificação de tumores, sendo os resultados obtidos também promissores.

Brown et al. (1999), por sua vez, utilizaram as SVMs sobre dados de *DNA microarray* para classificação de genes. Experimentos sobre esses dados sugerem que genes com função similar possuem padrão de expressão semelhante. Nos estudos conduzidos as SVMs se mostraram superiores a outras técnicas de AM, tais como as Árvores de Decisão e o *Kernel Fisher Discriminant*[5].

8.4 Considerações Finais

As SVMs são aplicáveis a uma grande gama de problemas práticos. Este Capítulo apresentou soluções em três áreas distintas: visão computacional, reconhecimento de dígitos e Bionformática.

[5]O *Kernel Fisher Discriminant* é um método baseado na teoria Estatística de discriminante de Fisher e no uso de funções Kernel, semelhante ao realizado em relação às SVMs.

Os resultados alcançados pelas SVMs nestes campos demonstram a eficácia desta técnica, principalmente quando confrontada com grandes volumes de dados e padrões complexos (com grande dimensão). Outros métodos de AM, quando aplicados a situações práticas desse tipo são, em geral, ineficientes, apresentando problemas de generalização (*overfitting*).

Capítulo 9

Conclusão

Este relatório apresentou uma introdução às Máquinas de Vetores Suporte (*Support Vector Machines* - SVMs). Essa técnica se caracteriza por alcançar altas taxas de precisão e ser robusta diante de dados de grande dimensionalidade (padrões com muitos atributos).

Apesar da obtenção de bons desempenhos em áreas variadas através da utilização desta técnica, algumas ressalvas devem ser feitas:

- A velocidade de classificação das SVMs pode ser menor que, por exemplo, as de RNAs (LeCun et al., 1995). Um trabalho que apresenta propostas visando atacar este problema foi realizado por Burges and Schölkopf (1997).

- Outro problema das SVMs está no fato de que a complexidade computacional na busca de soluções é da ordem de n^2 a n^3, em que n é o número de exemplos de treinamento. Apesar de polinomial, este valor pode ser significativo para aplicações nas quais o volume de dados é grande. Porém, em geral as SVMs apresentam tempos de treinamento consideravelmente menores que vários algoritmos de AM.

- A forma como o conhecimento é adquirido e codificado pelos classificadores gerados por SVMs não é facilmente interpretável. Esta é uma desvantagem sobre técnicas de AM como as Árvores de Decisão, que organizam o conhecimento adquirido em uma estrutura diretamente interpretável.

Embora as SVMs possuam algumas deficiências, existem diversos estudos e propostas no sentido de minimizá-las. Estes fatores em conjunto com a vantagem de uma alta capacidade de generalização faz com que as SVMs sejam técnicas largamente exploradas em trabalhos atuais.

Referências Bibliográficas

Almeida, M. B., Braga, A. P., and Braga, J. P. (2000). SVM-KM: Speeding SVMs learning with a priori cluster selection and k-means. In *Proceedings of the VI Brazilian Symposium on Neural Networks (SBRN'00)*, pages 162–167. IEEE Computer Society Press.

Baldi, P. and Brunak, S. (1998). *Bioinformatics - The Machine Learning Approach*. The MIT Press.

Baranauskas, J. A. and Monard, M. C. (2000). Reviewing some machine learning concepts and methods. Technical Report 102, Istituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, `ftp://ftp.icmc.usp.br/pub/BIBLIOTECA/rel_tec/RT_102.ps.zip`.

Ben-Dor, A., Bruhn, L., Friedman, N., and Nachman, I. (2000). Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7:559–584.

Ben-Hur, A., ad H. T. Siegelmann, D. H., and Vapnik, V. N. (2000). A support vector clustering method. In *Proceedings of the International Conference on Pattern Recognition (ICPR'00)*, volume 2, pages 724–727.

Braga, A., Carvalho, A. C. P. L. F., and Ludemir, T. (2000). *Redes Neurais Artificiais: Teoria e Aplicações*. Editora LTC.

Brown, M., Grundy, W., Lin, D., Christianini, N., Sugnet, C., Jr, M., and Haussler, D. (1999). Support vector machine classification of microarray gene expression data. Technical Report UCSC-CRL 99-09, Department of Computer Science, University California Santa Cruz, Santa Cruz, CA.

Burges, C. C. and Schölkopf, B. (1997). Improving the accuracy and speed of support vector machines. *Advances in Neural Information Processing Systems*, 9:375–382.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. *Knowledge Discovery and Data Mining*, 2(2):1–43.

Campbell, C. (2000). An introduction to kernel methods. In Howlett, R. J. and Jain, L. C., editors, *Radial Basis Function Networks: Design and Applications*, pages 155–192. Springer Verlag.

Campbell, C. and Cristianini, N. (1998). Simple learning algorithms for training support vector machines. Technical report, University of Bristol, `citeseer.nj.nec.com/campbell98simple.html`.

Carrieras, X. and Pérez, G. (2001). A comparison between SVM implementations. `http://www.lsi.upc.es/~lluism/seminaris/svmimpl.ps.gz`.

Carvalho, B. P. R. (2002). Support vector machines - um estudo sobre técnicas de treinamento. Monografia, Universidade Federal de Minas Gerais, `http://www.litc.cpdee.ufmg.br/upload/periodicos/Internal_Monograph58.zip`.

Casley, D. (1992). Primer on molecular biology. Technical report, U. S. Department of Energy, Office of Health and Environmental Research.

Chang, C.-C., Hsu, C.-W., and Lin, C.-J. (2000). The analysis of decomposition methods for support vector machines. *IEEE Transactions on Neural Networks*, 11(4):1003–1008.

Chang, C.-C. and Lin, C.-J. (2003). LIBSVM: a library for support vector machines. `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Chapelle, O., Vapnik, V., Bousquet, O., and Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1-3):131–159.

Chapelle, O. and Vapnik, V. N. (2000). Model selection for support vector machines. In Solla, S. A., Leen, T. K., and Müller, K.-R., editors, *Advances in Neural Information Processing Systems*. The MIT Press.

Collobert, R. and Bengio, S. (2001). SVMTorch: Support vector machines for large-scale regression problems. *Journal of Machine Learning Research*, 1:143–160.

Cortes, C. and Vapnik, V. N. (1995). Support vector networks. *Machine Learning*, 20:273–296.

Cravem, M. W. and Shavlik, J. W. (1994). Machine learning approaches to gene recognition. *IEEE Expert*, 9(2):2–10.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press.

DeCoste, D. and Schölkopf, B. (2001). Training invariant support vector machines. *Machine Learning*, 46:161–190.

Diamantaras, K. I. and Kung, S. Y. (1996). *Principal Component Neural Network*. Wiley, New York.

Ding, C. H. Q. and Dubchak, I. (2001). Multi-class protein fold recognition using Support Vector Machines and Neural Networks. *Bioinformatics*, 4(17):349–358.

Dowling, K. (1996). CMU robotics FAQ. `http://www.frc.ri.cmu.edu/robotics-faq/`.

Fernandez, R. and Viennet, E. (1999). Face identification using support vector machines. In *Proceedings of the European Symposium on Artificial Neural Networks (ESANN99)*, pages 195–200, Brussels. D.-Facto Press.

Frieβ, T.-T., Cristianini, N., and Campbell, C. (1998). The kernel adatron: a fast and simple learning procedure for support vector machines. In *Proceedings of the Fifteenth International Conference on Machine Learning*, Madison, Wisconsin. Morgan Kaufmann.

Friedman, J. H. (1996). Another approach to polychotomous classification. Technical Report 18, Department of Statistics, Stanford University, `http://www-stat.stanford.edu/reports/friedman/poly.ps.Z`.

Furey, T., Cristianini, N., Duffy, N., Bednarski, D., Schummer, M., and Haussler, D. (2001). Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16:906–914.

Ge, X. (2001). Sequential minimal optimization for svm. `http://datalab.uci.edu/people/xge/svm/`.

Haykin, S. (1999). *Neural Networks - A Compreensive Foundation*. Prentice-Hall, New Jersey, 2 edition.

Hearst, M. A., Schölkopf, B., Dumais, S., Osuna, E., and Platt, J. (1998). Trends and controversies - support vector machines. *IEEE Intelligent Systems*, 13(4):18–28.

Herbrich, R., Groepel, T., and Obermayer, K. (1999). *Large Margin Rank Boundaries for Ordinal Regression*, pages 29–45. In Smola et al. (1999a).

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multi-class support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425.

IDA, 2002 (2002). Ida benchmark repository used in several boosting, KFD and SVM papers. `http://www.ida.first.gmd.de/~raetscg/data/benchmarks.html`.

Jaakola, T. and Haussler, D. (1999). Probabilistic kernel regression models. In *Proceedins of the 1999 Conference on AI and Statistics*. Morgan Kaufmann.

Joachims, T. (1998). Making large-scale support vector machine learning practical. In Advances in Kernel Methods: Support Vector Machines, Schölkopf, B., Burges, C., and Smola, A. (editors), MIT Press.

LeCun, Y. A., Jackel, L. D., Bottou, L., Brunot, A., Cortes, C., Denker, J. S., Drucker, H., Guyon, I., Müller, E., Säckinger, E., Simard, P. Y., and Vapnik, V. N. (1995). Comparison of learning algorithms for handwritten digit recognition. In Foulgeman-Soulié, F. and Gallinari, P., editors, *Proceedings of the International Conference on Artificial Neural Networks (ICANN´95)*, volume 2, pages 53–60, Nanterre, France.

Lorena, A. C., Batista, G. E. A. P. A., de Carvalho, A. C. P. L. F., and Monard, M. C. (2002a). The influence of noisy patterns in the performance of learning methods in the splice junction recognition problem. In *IEEE Proceedings of the VII Brazilian Symposium on Neural Networks (SBRN 2002)*, pages 31–36. IEEE Computer Society Press.

Lorena, A. C., Batista, G. E. A. P. A., de Carvalho, A. C. P. L. F., and Monard, M. C. (2002b). Splice junction recognition using machine learning techniques. In Bazzan, A. L. C. and de Carvalho, A. C. P. L. F., editors, *Proceedings of the First Brazilian Workshop on Bioinformatics*, pages 32–39.

Mayoraz, E. and Alpaydm, E. (1998). Support vector machines for multi-class classification. Research Report IDIAP-RR-98-06, Dalle Molle Institute for Perceptual Artificial Intelligence, Martigny, Switzerland.

Mercer, J. (1909). Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, A 209:415–446.

Mitchell, T. (1997). *Machine Learning*. McGraw Hill.

Müller, K. R., Mika, S., Rätsch, G., Tsuda, K., and Schölkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–201.

Murtagh, B. A. and Saunders, M. A. (1993). Minos 5.4 user's guide. Technical Report SOL 83.20, Stanford University.

Osuna, E., Freund, R., and Girosi, F. (1997). An improved training algorithm for support vector machines. In Principe, J., Gile, L., Morgan, N., and Wilson, E., editors, *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing VII*, pages 276–285, New York. IEEE Computer Society.

Platt, J. (1999). Fast training of support vector machines using sequential minimal optimization. In Schölkopf, B., Burges, C. J. C., and Smola, A. J., editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208, Cambridge, MA. The MIT Press.

Platt, J. C., Cristiani, N., and Shawe-Taylor, J. (2000). Large margin DAGS for multiclass classification. In *Advances in Neural Information and Processing Systems*, pages 547–553. The MIT Press.

Russel and Norvig (1995). *Artificial Intelligence - a Modern Approach*. Prentice Hall.

Schapire, R., Freund, Y., Barlett, P., and Lee, W. S. (1997). Boosting the margin: a new explanation for the effectiveness of voting methods. In Jr., D. H. F., editor, *Proceedings of the International Conference on Machine Learning (ICML97)*, pages 322–330. Morgan Kaufmann.

Schölkopf, B., Simard, O., Smola, A., and Vapnik, V. (1998). Prior knowledge in support vector kernels. In Jordan, M. I., Kearns, M. J., and Solla, S. A., editors, *Advances in Neural Information and Processing Systems*, pages 640–646. The MIT Press.

Schwaighofer, A. (2002). SVM toolbox for Matlab. `http://www.cis.tugraz.at/igi/aschwaig/software.html`.

Sewell, M. (2001). mySVM. `http://www.cs.ucl.ae.uk/staff/M.Sewell/SVM`.

Smola, A. and Schölkopf, B. (2000). Kernel machines homepage. `www.kernel-machines.org`.

Smola, A. J., Barlett, P., Schölkopf, B., and Schuurmans, D. (1999a). *Advances in Large Margin Classifiers*. MIT Press.

Smola, A. J., Barlett, P., Schölkopf, B., and Schuurmans, D. (1999b). *Introduction to Large Margin Classifiers*, chapter 1, pages 1–28. In Smola et al. (1999a).

Smola, A. J. and Schölkopf, B. (2002). *Learning with Kernels*. The MIT Press, Cambridge, MA.

Sung, K. and Poggio, T. (1994). Example-based learning for view-based human face detection. *A. I. Memo 1521, C. B. C. L. paper 112*.

Tamayo, P., Slonim, D., Mesirov, J., Zhum, Q., Kitareewan, S., Dmitrovsky, E., Lander, E., and Golub, T. (1996). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. In *PNAS*, pages 2907–2912.

Tay, F. E. H. and Cao, L. (2001). Application of support vector machines in financial time series forecasting. *Omega, The International Journal of Management Science*, 29:309–317.

Vanderbei, R. (1994). LOQO: an interior point code for quadratic programming. Technical Report SOR 94-15, Princeton University.

Vapnik, V. N. (1982). *Estimation of dependencies based on empirical data*. Springer-Verlag, New York.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag.

Vert, J.-P. (2001). Introduction to support vector machines and applications to computational biology (draft). `http://web.kuicr.kyoto-u.ac.jp/~vert/research/semsvm/`.

Wahba, G. (2000). An introduction to model building with reproducing kernel hilbert spaces. Technical Report 1020, Statistics Department of University of Wisconsin-Madison.

Weston, J. and Watkins, V. (1998). Multi-class support vector machines. Technical Report CSD-TR-98-04, Department of Computer Science, University of London.

Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengaeuer, T., and Müller, K. R. (2000). Engineering suppot vector machine kernels that recognize translation initiation sites in DNA. *Bioinformatics*, 16:906–914.

Apêndice A

Teoria de Otimização

Este Apêndice apresenta conceitos básicos em Teoria de Otimização. Esta descrição é simplificada, formando uma base para o entendimento dos princípios apresentados na determinação do hiperplano ótimo para as SVMs. As conceituações apresentadas foram extraídas de Vert (2001) e (Cristianini and Shawe-Taylor, 2000).

A.1 Otimização de função sem restrições

Considerando um problema de minimização, o caso mais simples consiste em encontrar o ponto $\mathbf{x}^*$ (em que $\mathbf{x} \in \Re^m$) que minimize uma função $f(\mathbf{x})$ convexa, sem restrições no valor procurado. Considera-se apenas funções convexas pelo fato deste ser o tipo de função minimizada pelas SVMs.

Sendo $\nabla f(\mathbf{x})$ o vetor de derivadas parciais de f (derivadas em relação a cada coordenada de $\mathbf{x}$), ou vetor gradiente (Equação A.1), o ponto ótimo $\mathbf{x}^*$ é encontrado igualando-se $\nabla f(\mathbf{x})$ a 0. Resolve-se então o conjunto de Equações A.2.

$$\nabla f(\mathbf{x}) = \left(\frac{\partial f}{\partial w_1}(\mathbf{x}), \ \ldots, \ \frac{\partial f}{\partial w_m}(\mathbf{x}) \right) \tag{A.1}$$

$$\begin{cases} \partial f(\mathbf{x}) / \partial x_1 = 0 \\ \partial f(\mathbf{x}) / \partial x_2 = 0 \\ \vdots \\ \partial f(\mathbf{x}) / \partial x_n = 0 \end{cases} \tag{A.2}$$

A.2 Otimização de função com uma restrição

O problema atacado neste caso é representado por:

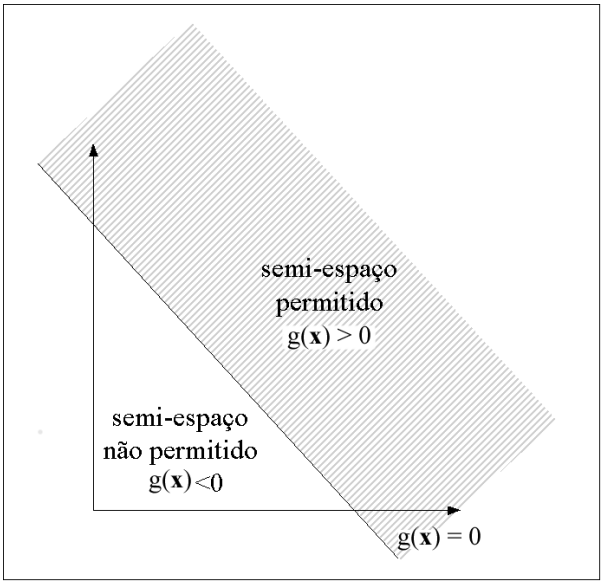

Figura A.1: Ilustração do semi-espaço de possíveis soluções ao problema de otimização com uma restrição linear.

$$\textbf{Minimizar: } f(\mathbf{x})$$
$$\textbf{Sob a restrição: } g(\mathbf{x}) \geq 0$$

Se a restrição for linear, como no caso das SVMs, o conjunto de vetores $\mathbf{x}$ que satisfazem a restrição pertencem a um semi-espaço delimitado pelo hiperplano $g(\mathbf{x}) = 0$. A Figura A.1 ilustra este semi-espaço, o qual é denominado por simplicidade de referência "semi-espaço permitido". O semi-espaço que não obedece à restrição será referenciado por "semi-espaço não-permitido".

Deve-se procurar um ponto $\mathbf{x}^*$ pertencente ao semi-espaço permitido com $f(\mathbf{x}^*)$ mínimo.

Seja $\mathbf{x}$ o ponto ótimo procurado (que minimiza $f(\mathbf{x})$ e obedece a restrição) e $\mathbf{x}_0^*$ o ponto que minimiza $f(\mathbf{x})$, sem restrição. Considerando que a função $g(\mathbf{x})$ é da forma $\mathbf{w} \cdot \mathbf{x} + b = 0$ (linear), duas situações podem ocorrer em relação ao posicionamento de $\mathbf{x}_0^*$:

(a) $\mathbf{x}_0^*$ pertence ao semi-espaço permitido. Neste caso, $\mathbf{x}_0^*$ é o valor ótimo procurado, ou seja, $\mathbf{x}^* = \mathbf{x}_0^*$. Logo, $\mathbf{x}^*$ satisfaz as igualdades A.3.

$$\begin{cases} \nabla f(\mathbf{x}^*) = 0 \\ \mathbf{w} \cdot \mathbf{x}^* + b \geq 0 \end{cases} \tag{A.3}$$

O ponto ótimo $\mathbf{x}^*$ pode então ser encontrado pelo sistema A.3.

(b) $\mathbf{x}_0^*$ pertence ao semi-espaço não permitido, ou seja, não obedece à restrição. Neste caso, é provado matematicamente que $\mathbf{x}^*$ encontra-se sobre o hiperplano $\mathbf{w} \cdot \mathbf{x} + b = 0$ e que o gradiente $\nabla f(\mathbf{x})$ é paralelo a $\mathbf{w}$. O problema de otimização passa a ser encontrar um ponto $\mathbf{x}^*$ que satisfaça o sistema A.4.

$$\begin{cases} \nabla f(\mathbf{x}^*) = \alpha \mathbf{w}, \text{ para algum real } \alpha \\ \mathbf{w} \cdot \mathbf{x}^* + b = 0 \end{cases} \tag{A.4}$$

O ponto ótimo $\mathbf{x}^*$ deve satisfazer então os sistemas A.3 ou A.4. Reunindo estes dois resultados, chega-se ao sistema A.5, para algum real α.

$$\begin{cases} \nabla f(\mathbf{x}^*) = \alpha \mathbf{w} \\ \mathbf{w} \cdot \mathbf{x}^* + b > 0 \\ \alpha \geq 0 \\ \alpha (\mathbf{w} \cdot \mathbf{x}^* + b) = 0 \end{cases} \tag{A.5}$$

A.3 Otimização com várias restrições

Neste caso, tem-se o problema:

$$\textbf{Minimizar: } f(\mathbf{x})$$
$$\textbf{Sob as restrições: } g_i(\mathbf{x}) \geq 0, \text{ para } i = 1, \ \ldots, \ n$$

Similarmente ao apresentado na Seção anterior, cada restrição define um semi-espaço permitido e um não permitido, delimitados por $g_i(\mathbf{x}) = 0$. A intersecção dos semi-espaços permitidos representa a região onde a solução ótima deve ser procurada.

Para se solucionar este problema, introduz-se uma função denominada Lagrangiana, que incorpora informações da função objetivo e das restrições a serem obedecidas[1]. A Lagrangiana para o problema apresentado é dada pela função objetivo subtraída[2] de uma combinação linear das restrições, conforme representado na Equação A.6. Os coeficientes da combinação (α_i) são usualmente denominados multiplicadores de Lagrange.

$$L(\mathbf{x}, \alpha) = f(\mathbf{x}) - \sum_{i=1}^{n} \alpha_i g_i(\mathbf{x}) \tag{A.6}$$

[1] A teoria de Lagrange também é utilizada nos casos sem restrição e com uma restrição. Optou-se por formalizá-la somente no caso de mais restrições para facilitar o entendimento dos conteúdos apresentados.

[2] Se a restrição fosse $g(\mathbf{x}) \leq 0$, a diferença seria substituída por uma soma.

A obtenção de uma solução ótima $(\mathbf{x}^*, \alpha^*)$ neste caso é dada pelo sistema A.7, que é uma generalização dos resultados apresentados em A.5. Esta solução faz parte do Teorema de Kuhn-Tucker.

$$\begin{aligned} &\partial L\left(\mathbf{x}^*, \alpha^*\right)/_{\partial \mathbf{X}} = 0 \\ &\alpha_i^* g_i\left(\mathbf{x}^*\right) = 0, \;\; i = 1, \ldots, n \\ &g_i\left(\mathbf{x}^*\right) \geq 0, \;\; i = 1, \ldots, n \\ &\alpha_i^* \geq 0, \;\; i = 1, \ldots, n \end{aligned} \tag{A.7}$$

A Equação A.6 deve ser maximizada em relação a α e minimizada em relação a $\mathbf{w}$, com a restrição das coordenadas do vetor α serem maiores que 0. Esta é a representação *primal* do problema de otimização. O cálculo da derivada primeira da Lagrangiana em relação as variáveis primais leva a um conjunto de equações. Substituindo os resultados dessas equações da Lagrangiana e em suas restrições, obtém-se a representação *dual* do problema. A função resultante contêm somente os multiplicadores de Lagrange α_i (também denominados variáveis duais) e deve ser maximizada sob restrições mais simples. A formulação dual em geral é mais simples de ser solucionada, já que frequentemente é difícil lidar com as restrições de desigualdade do problema primal diretamente. Obtidos os valores dos α_i^* por algum algoritmo de otimização padrão (normalmente numérico), as variáveis primais ótimas podem ser facilmente determinadas.

O exemplo a seguir ilustra os conceitos apresentados (Cristianini and Shawe-Taylor, 2000).

Exemplo: Dado o problema de minimização com função objetivo quadrática:

$$\begin{gathered} \text{Minimizar } \tfrac{1}{2}\mathbf{w}'\mathbf{Q}\mathbf{w} - \mathbf{k}'\mathbf{w} \\ \text{Sujeito a } \mathbf{X}\mathbf{w} \leq \mathbf{c} \end{gathered}$$

em que $\mathbf{Q}$ é uma matriz positivamente definida (com auto-valores maiores que 0) de dimensão n x n, $\mathbf{k}$ é um vetor n-dimensional, $\mathbf{c}$ é um vetor m-dimensional e $\mathbf{X}$ é uma matriz de dimensão n x n.

Este é um caso de programa quadrático, mesma classe de problemas das SVMs. A Lagragiana deste problema é dada pela Equação A.8.

$$L(\mathbf{w}, \alpha) = \frac{1}{2}\mathbf{w}'\mathbf{Q}\mathbf{w} - \mathbf{k}'\mathbf{w} + \alpha'\left(\mathbf{X}\mathbf{w} - \mathbf{c}\right) \tag{A.8}$$

Esta função deve ser minimizada em relação a $\mathbf{w}$ e maximizada em relação a α, com a restrição $\alpha \geq 0$. Esta é a representação primal do problema.

Derivando a Equação A.8 em relação a $\mathbf{w}$ e igualando o resultado a 0, obtém-se o resultado A.9.

$$\mathbf{w} = \mathbf{Q}^{-1}(\mathbf{k} - \mathbf{X}'\alpha) \tag{A.9}$$

Substituindo este valor no problema primal, tem-se a representação dual:

$$\begin{gathered} \text{Maximizar } -\tfrac{1}{2}\alpha'\mathbf{P}\alpha - \alpha'\mathbf{d} - \tfrac{1}{2}\mathbf{k}'\mathbf{Q}\mathbf{k} \\ \text{Sujeito a } \alpha \geq 0 \end{gathered}$$

onde $\mathbf{P} = \mathbf{X}\mathbf{Q}^{-1}\mathbf{X}'$ e $\mathbf{d} = \mathbf{c} - \mathbf{X}\mathbf{Q}^{-1}\mathbf{k}$.

Logo, o dual de um programa quadrático é outro programa quadrático, porém com restrições mais simples. Este problema pode ser então solucionado por algoritmos de otimização. Determinado o valor ótimo do vetor α^*, $\mathbf{w}^*$ pode ser determinado pela Equação A.9.
