UNIVERSIDADE ESTADUAL DE CAMPINAS

Faculdade de Engenharia Elétrica e de Computação

Comitê de Máquinas: Uma Abordagem Unificada Empregando Máquinas de Vetores-Suporte

Clodoaldo Aparecido de Moraes Lima

Orientador: Prof. Dr. Fernando José Von Zuben

Tese apresentada à Pós-graduação da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como requisito parcial à obtenção do Título de **Doutor em Engenharia Elétrica** na área de Engenharia de Computação.

Banca Examinadora:

Prof. Dr. Maurício Fernandes Figueiredo – DIN/UEM

Prof. Dr. Walmir Matos Caminhas – DEE/UFMG

Prof. Dr. Márcio Luiz de Andrade Netto – FEEC/Unicamp

Prof. Dr. Paulo Augusto Valente Ferreira – FEEC/Unicamp

Prof. Dr. Wagner Caradori do Amaral – FEEC/Unicamp

Campinas, 10 dezembro de 2004

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Lima, Clodoaldo Aparecido de Moraes
L628c Comitê de máquinas: uma abordagem unificada empregando máquinas de vetores-suporte / Clodoaldo Aparecido de Moraes Lima. --Campinas, SP: [s.n.], 2004.

Orientador: Fernando José Von Zuben
Tese (doutorado) - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Kernel, funções de. 2. Sistema especialistas (Computação). 3. Redes neurais (computação). 4. Inteligência Artificial. 5. Aprendizado do computador. 6. Teoria de modelo. 7. Teoria da aproximação. I. Zuben, Fernando José Von. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Titulo em Inglês: Committee machines: a unified approach using support vector machines
Palavras-chave em Inglês: Kernel Functions, Expert Systems (Computer Science), Neural Networks (Computer science), Artificial Intelligence, Machine Learning, Model Theory e Approximation Theory.
Área de concentração: Engenharia de Computação
Titulação: Doutor em Engenharia Elétrica
Banca examinadora: Mauricio Fernandes Figueiredo, Walmir Matos Caminhas, Márcio Luiz de Andrade Netto, Paulo Augusto Valente Ferreira e Wagner Caradori do Amaral.
Data da defesa: 10/12/2004

Resumo

Os algoritmos baseados em métodos de *kernel* destacam-se entre as diversas técnicas de aprendizado de máquina. Eles foram inicialmente empregados na implementação de máquinas de vetores-suporte (SVMs). A abordagem SVM representa um procedimento de aprendizado não-paramétrico para classificação e regressão de alto desempenho. No entanto, existem aspectos estruturais e paramétricos de projeto que podem conduzir a uma degradação de desempenho. Na ausência de uma metodologia sistemática e de baixo custo para a proposição de modelos computacionais otimamente especificados, os comitês de máquinas se apresentam como alternativas promissoras. Existem versões estáticas de comitês, na forma de ensembles de componentes, e versões dinâmicas, na forma de misturas de especialistas. Neste estudo, os componentes de um ensemble e os especialistas de uma mistura são tomados como SVMs. O objetivo é explorar conjuntamente potencialidades advindas de SVM e comitê de máquinas, adotando uma formulação unificada. Várias extensões e novas configurações de comitês de máquinas são propostas, com análises comparativas que indicam ganho significativo de desempenho frente a outras propostas de aprendizado de máquina comumente adotadas para classificação e regressão.

Abstract

Algorithms based on kernel methods are prominent techniques among the available approaches for machine learning. They were initially applied to implement support vector machines (SVMs). The SVM approach represents a nonparametric learning procedure devoted to high performance classification and regression tasks. However, structural and parametric aspects of the design may guide to performance degradation. In the absence of a systematic and low-cost methodology for the proposition of optimally specified computational models, committee machines emerge as promising alternatives. There exist static versions of committees, in the form of ensembles of components, and dynamic versions, in the form of mixtures of experts. In the present investigation, the components of an ensemble and the experts of a mixture are taken as SVMs. The aim is to jointly explore the potentialities of both SVM and committee machine, by means of a unified formulation. Several extensions and new configurations of committee machines are proposed, with comparative analyses that indicate significant gain in performance before other proposals for machine learning commonly adopted for classification and regression.

Dedicatória

Aos **meus pais**, pelo amor, apoio e dedicação constantes, durante toda a minha vida, sobretudo por terem me ensinado que a fé é o fundamento de nossa existência. Se hoje a minha vida se enriquece e se abre à ciência, isso se deve aos seus ensinamentos, os quais me nutrem para que eu não perca o entusiasmo na busca do saber.

Aos **meus irmãos Fabrício e Rivelino**, pelo estímulo, carinho e exemplo no exercício dos ensinamentos paternos. Por terem compartilhado de minhas aspirações, eles compartilham comigo a satisfação de chegar ao ideal sonhado.

À **minha Jeniffer**, amiga, companheira e esposa, pelo incentivo, disponibilidade irrestrita e presença constante ao longo da elaboração desta tese. Ainda, pela maneira carinhosa e compreensiva com que sempre me apoiou. Parceira em todos os momentos e onipresente em meus passos vindouros.

De todo o coração, dedico este trabalho a vocês.

Agradecimentos

Meu Agradecimento

Ao Professor Doutor Fernando José Von Zuben, orientador deste trabalho, pela amizade, pelo apoio nos momentos difíceis e pela confiança em mim depositada quando aceitou ser o meu orientador. Acima de tudo, pela presença humana e científica marcante, capaz de agregar esforços e somar capacidades – sem risco de associação com iniciativas análogas adotadas pelas estratégias de aprendizado de máquina que são tema da tese –, preenchendo assim o modelo de mestre e orientador. O entusiasmo do Professor Fernando nos leva a vasculhar terrenos inexplorados e proporciona novas formas de investigação daquilo que parecia débil e esgotado. Tocado por este entusiasmo, jamais serei o mesmo; impulsionado por este exemplo, tentarei multiplicá-lo; e em cada passo de minha atuação profissional, e também fora dela, pretendo repetir e levar adiante a postura intelectual e a gratuidade de propósitos que regem as suas atitudes.

Meu sincero agradecimento pela orientação, mais uma vez, precisa, experiente, sincera e amiga durante a elaboração deste trabalho. Pela dedicação e competência demonstradas, que o torna um modelo a ser seguido por todos.

A todos os amigos que conviveram comigo no **Laboratório de Bioinformática e Computação Bio-Inspirada (LBiC)**, pelo clima de companheirismo e respeito mútuo, requisitos importantes para a criação de condições de trabalho e pesquisa adequadas.

Aos **Professores da Faculdade de Engenharia Elétrica e de Computação (FEEC) da Unicamp, particularmente aos membros do Departamento de Engenharia de Computação e Automação Industrial (DCA)**, pela amizade e cordialidade que sempre me dispensaram, e por tudo que me ensinaram, contribuindo em muito para o meu aprimoramento profissional.

À **Faculdade de Engenharia Elétrica e de Computação (FEEC) da Unicamp**, pela oportunidade de realizar o curso de doutoramento, e aos responsáveis pela **chefia do Departamento de Engenharia de Computação e Automação Industrial (DCA)** durante o período de desenvolvimento deste trabalho, por permitirem acesso às instalações.

Ao **Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq)**, pelo suporte financeiro, sob processo no. 141394/2000-5, sem o qual não seria possível o desenvolvimento da pesquisa. São iniciativas como estas que efetivamente contribuem para o desenvolvimento científico e tecnológico de um país.

Agradecimentos Especiais

Meu agradecimento especial

Ao **meu amigo Wilfredo Jaime Puma Villanueva**, para o qual não vêm palavras para expressar toda a minha gratidão, mas vou tentar: obrigado pela ajuda, pelas discussões, pelo apoio, estímulo, compreensão e por tantas outras coisas que, por vezes, somente bons amigos podem fazer por nós.

Ao **meu amigo André Luiz Vasconcelos Coelho**, mente aberta e formação eclética, pelas discussões e participação ativa em etapas relevantes deste trabalho, pelo arejamento intelectual proporcionado e pelo agradável convívio nestes anos.

Ao **meu amigo Eurípedes Pinheiro dos Santos**, pelos questionamentos indispensáveis e pelas críticas construtivas, ambos fundamentais para a sedimentação e amadurecimento das linhas de argumentação ao longo do texto da tese. Ainda, pelas sugestões de material bibliográfico e ajuda na implementação dos vários algoritmos utilizados nesta tese.

Ao **meu amigo Moisés Vidal Ribeiro**, que mais do que ninguém acompanhou os passos iniciais deste trabalho, dentro e fora do ambiente acadêmico, pela convivência fraterna e por transmitir serenidade e valores éticos a todos à sua volta.

Aos **demais amigos e companheiros** que fazem do LBiC um ambiente saudável e motivador para a pesquisa, em especial àqueles que gentilmente e zelosamente contribuíram junto à revisão e correção de versões preliminares de parte do texto desta tese.

A todos os amigos que, direta ou indiretamente, contribuíram para a realização desta conquista,

o meu muito obrigado!

Sumário

Lista de Abreviações

Adaboost	*Ada*ptive *Boost*ing
Adaboost.R	Adaboost para problema de regressão
Bagging	*B*ootstrap *ag*regat*ing*
BCD	Problema de diagnóstico de câncer de cólon
BC-SVMs	Comitê Bayesiano de SVMs
Boosting	Usado para descrever técnicas que *impulsionam* o desempenho de classificadores ou regressores ao agregá-los
Boost-SMO	Aplicação do algoritmo boosting juntamente com o algoritmo SMO
DAGSVM	Grafo Acíclico Direcionado aplicado a Máquinas de Vetores-Suporte
DIAB	Problema de diabetes
EM	Algoritmo de Maximização da Esperança
EME-SVM	Ensemble com múltiplos estágios para SVM
EME-SVMc	Ensemble com múltiplos estágios para SVM utilizando um método construtivo
EME-SVMp	Ensemble com múltiplos estágios para SVM utilizando um método de poda
EQM	Erro Quadrático Médio
ERBF	Função de base radial exponencial
ERM	Princípio de Minimização do Risco Empírico
GASel	Critério de seleção de modelos baseada em algoritmo genético
GEM	Algoritmo de Maximização da Esperança Generalizada
GLIM	Modelo linear generalizado

GME	Mistura de especialistas *gated* – onde a rede *gating* e/ou a rede especialista é uma Perceptron Multicamada
HEP	Problema de hepatite
HE-SVM-B	Ensemble de SVMs Heterogêneas no nível de classificação Binária
HE-SVM-M	Ensemble de SVMs Heterogêneas no nível de classificação para múltiplas classes
HE-SVMs	Ensemble de SVMs Heterogêneas
HMEs	Mistura Hierárquica de Especialistas
IQR	Intervalo Interquartil
IRLS	Método Iterativo para Quadrados Mínimos Recursivos
IWLS	Método Iterativo para Quadrados Mínimos Ponderado
KKT	Condições de Karush-Khun-Tucker
LA	Linhas de atraso
LME	Mistura de Especialista Localizados – onde o modelo para uma rede *gating* é um kernel normalizado
LMS	Média dos Quadrados Mínimos
LRU	Menos recentemente utilizado
LS	Quadrados Mínimos
LS-SVM	Máquina de Vetores-Suporte Baseada em Quadrados Mínimos
Max Wins	Número máximo de vitórias
MDL	Princípio do Comprimento da Descrição Mínima
ME[SVM]	Mistura de Especialistas baseado em Máquinas de Vetores Suporte
MEs	Mistura de Especialistas
MLP	Perceptron multicamada
nox	Nível de óxido nitroso no problema de *Housing Boston*

NSVs	Número de Vetores-Suporte
OCR	Reconhecimento Óptico de Caracteres
OLC	Combinação Linear Ótima
PAC	PAC *Learning* – Aprendizado Provavelmente Aproximadamente Correto
PoG	Porcentagem de Ganho
price	Preço médio das casas no problema de *Housing Boston*
PSel	Critério de seleção de modelos baseada em PERRONE & COOPER (1993)
QP	Programação Quadrática
RBF	Função de Base Radial
RFE	Algoritmo de Eliminação Recursiva de Características
RKHS	Espaço de reprodução de *kernel* de Hilbert
SD	Desvio Padrão
SMO	Algoritmo de Otimização de Seqüência Mínima
SRM	Princípio de Minimização do Risco Estrutural
SSE	Soma dos Quadrados dos Erros
S-SVM	Máquinas de Vetores-Suporte isolada (não participante de um comitê de máquinas)
SVC	Classificação por Vetores-Suporte
SVMs	Máquinas de Vetores-Suporte
SVR	Regressão por Vetores-Suporte
SVs	Vetores-Suporte
VC	VC *dimension* – Dimensão de Vapnik-Chervonenkis
VCSel	Seleção de modelos baseado na dimensão VC
WDBC	Problema de câncer da Universidade de Wisconsin

ZSel	Critério de seleção de modelos baseada em ZHOU (2002)

Lista com os Principais Símbolos

b	É o termo de polarização ou bias
C	Estabelece o compromisso entre a complexidade do modelo e o erro de treinamento
d	É a dimensão dos dados de saída
h	É a dimensão VC
$\boldsymbol{H}$	Espaço de hipóteses
K	Número de classes no conjunto de treinamento
m	Número de especialistas
M	Número de classificadores ou regressores em um ensemble
n	Dimensão do espaço de entrada
N	Número de amostras do conjunto de treinamento
R	É o raio de uma hiper-esfera circundando todas as amostras de um conjunto de dados
$\mathbf{w}$	É o vetor de pesos do classificador ou regressor
$\mathbf{X}$	É uma matriz com os dados de entrada de dimensão $N \times n$
$\mathbf{x}_i$	É *o* i-ésimo vetor de entrada
x_{ij}	É o j-ésimo componente do i-ésimo vetor de entrada

$\mathbf{Y}$	É uma matriz com os dados de saída desejado de dimensão $N \times d$
$\mathbf{y}_i$	É o *i*-ésimo vetor de saída desejado
$K(\mathbf{x}_i, \mathbf{x}_j)$	É a função *kernel* (veja mais detalhes no capítulo 4, seção 4.5), que vai representar, no espaço original, um produto interno realizado no espaço de características
$P(\mathbf{y} \mid \mathbf{x}, \theta_i)$	É a probabilidade do especialista *i* gerar a saída **y** baseado na entrada **x** e no vetor de parâmetros θ_i
$P(i \mid \mathbf{x}, \mathbf{v}^0)$	É a probabilidade de se escolher o especialista *i*, dados a entrada **x** e o vetor de parâmetros $\mathbf{v}^0$
$h_i^{(t)}$	Para uma mistura de especialistas, é a probabilidade *a posteriori* do *i*-ésimo especialista. Para uma mistura hierárquica de especialistas, é a probabilidade *a posteriori* e pode ser vista como o crédito atribuído ao *i*-ésimo nó não-terminal da rede hierárquica.
$g_i(\mathbf{x}, \mathbf{v})$	É a saída da rede *gating*. Pode ser interpretada como a probabilidade a priori, ou seja, a probabilidade da *gating* escolher o *i*-ésimo especialista, dada somente a entrada **x**.
μ_i	É a saída do especialista *i*
θ_i	É o vetor de parâmetros do especialista *i*
$\alpha_i (i = 1, \ldots, N)$	São os multiplicadores de Lagrange
θ_j^{k+1}	Representa os parâmetros do *j*-ésimo especialista para a iteração *k*+1
σ	Variância do *kernel* RBF (Funções de Base Radial)

ρ(w,b)	Margem de separação dos hiperplanos
ε	É a precisão desejada na formulação de Máquinas de Vetores-Suporte para regressão
$l(\Theta;\chi)$	É a função de verossimilhança
$Q(\theta,\theta^{(k)})$	É o valor esperado da função de verossimilhança
$\boldsymbol{\chi} = \{(\mathbf{x}^{(t)},\mathbf{y}^{(t)})\}_{t=1}^{N}$	Conjunto de dados de treinamento
h_i	A probabilidade *a posteriori* do nó de uma rede hierárquica. Pode ser vista como o crédito atribuído ao *i*-ésimo nó não-terminal da rede.
$h_{j\|i}$	É a probabilidade *a posteriori* do ramo para o caso de mistura hierárquica de especialistas. Pode ser vista como o crédito atribuído aos ramos acima dos nós não-terminais.
h_{ij}	É a probabilidade *a posteriori* do especialista (*i*,*j*) e pode ser vista como o crédito atribuído ao especialista (*i*,*j*).

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Posicionamento e motivação da pesquisa

Aprendizado de máquina envolve a automação da manipulação de informação e da representação do conhecimento (LANGLEY & SIMON, 1995). As fronteiras de aplicação de tais metodologias computacionais vêm se expandindo rapidamente ao longo dos últimos anos, e esta expansão é patrocinada por técnicas e conceitos derivados da estatística, da matemática aplicada e da ciência da computação. É neste cenário que se insere a síntese de ferramentas computacionais para treinamento supervisionado e não-supervisionado, a partir de dados amostrados.

Sob várias perspectivas, os últimos anos têm assistido uma mudança de paradigma na área de aprendizado de máquina, na seqüência daquela ocorrida em meados da década de 1981-1990, quando foram propostos algoritmos voltados para a síntese de aproximadores universais de funções não-lineares contínuas, definidas em regiões compactas do espaço de aproximação (HORNIK *et al.*, 1989). Merecem destaque aqui as redes neurais artificiais e suas extensões (HAYKIN, 1999).

No entanto, a capacidade de aproximação universal, geralmente vinculada a modelos matemáticos semi-paramétricos e não-paramétricos (HÄRDLE, 1990), é uma propriedade existencial. Sua aplicação junto a problemas práticos requer um controle da flexibilidade do modelo de aproximação, de modo que o grau de flexibilidade deste modelo

seja compatível (nem muito maior, nem muito menor) com a complexidade intrínseca do problema a ser tratado.

Em treinamento supervisionado e não-supervisionado a partir de dados amostrados, esta exploração consistente da capacidade de aproximação universal produz o que se convencionou chamar de maximização da capacidade de generalização, ou seja, obtenção de desempenho ótimo junto a dados não observados durante o processo de treinamento. A teoria de regularização (GIROSI *et al.*, 1995), assim como conceitos de teoria de informação (MACKAY, 2003) e de teoria de aprendizado estatístico (HASTIE *et al.*, 2001) fornecem subsídios para operar de forma eficaz com modelagem não-paramétrica em aprendizado de máquina.

Mas foi com o advento das máquinas de vetores-suporte (SVM's, do inglês *support vector machines*) (VAPNIK, 1995) e outros métodos de *kernel* (SCHÖLKOPF & SMOLA, 2002) que a maximização da capacidade de generalização de aproximadores universais passou a ser embasada em resultados teóricos de aplicação imediata.

Por operarem no espaço original dos dados, em que as não-linearidades presentes e a complexidade intrínseca do problema não são conhecidas a priori, as ferramentas anteriormente propostas para aprendizado de máquina e que apresentam capacidade de aproximação universal, como redes neurais artificiais, podem induzir modelos matemáticos com baixa capacidade de generalização. Esta é a razão pela qual as máquinas de vetores-suporte são classificadas por alguns autores (CRISTIANINI & SCHÖLKOPF, 2002) como a nova geração de algoritmos de aprendizado, pois tendem a operar em um espaço de dimensão bem maior que a dimensão do espaço original, denominado espaço de características. Neste espaço de características, problemas de otimização quadrática com restrições são então formulados visando maximizar explicitamente a capacidade de generalização.

O espaço de características é sustentado por uma teoria geral de aprendizado de máquina, que abre a possibilidade de se operar com novas representações eficientes de funções não-lineares, empregadas no projeto de algoritmos de aprendizado. Estas representações fazem uso das chamadas funções *kernel*, que permitem representar no espaço original produtos escalares realizados no espaço de características (VAPNIK, 1995).

O processo de treinamento supervisionado ou não-supervisionado vai estar agora associado a um problema de otimização convexa, não sendo assim susceptível a mínimos locais. Além disso, a complexidade do modelo matemático independe da dimensão do espaço de entrada e será definida de acordo com uma estimativa da complexidade intrínseca do problema, a partir dos dados disponíveis.

No entanto, existem aspectos estruturais e paramétricos de projeto que podem conduzir a uma degradação de desempenho das máquinas de vetores-suporte. Isto implica que as garantias teóricas de maximização da capacidade de generalização podem ser drasticamente violadas, pela simples atribuição equivocada de algum parâmetro (ou um subconjunto de parâmetros) de projeto. Dentre os parâmetros de projeto, se encontram: (*i*) o tipo de função *kernel*; (*ii*) parâmetros da função *kernel* adotada; (*iii*) parâmetros do problema de otimização quadrática com restrições.

Em paralelo à crescente difusão de técnicas e modelos de aprendizado de máquina baseados em funções *kernel*, aparecem os comitês de máquinas (HAYKIN, 1999). Como o próprio nome indica, um comitê de máquinas representa a agregação de mais de uma máquina de aprendizado na produção de uma única solução computacional para um determinado problema. Além de estarem motivados pela mesma busca da maximização da capacidade de generalização, os comitês de máquinas são motivados por pelo menos outras duas razões:

- grande disponibilidade de recursos computacionais para possibilitar a síntese de múltiplas propostas de soluções para todo ou parte do problema sendo tratado;
- a demonstração, através do teorema "*no free lunch*" (WOLPERT & MACREADY, 1996), de que não existem modelos genéricos de aprendizado de máquina que, em média, apresentem melhor desempenho que qualquer outro modelo para uma classe de problemas quaisquer.

Sendo assim, na ausência de uma metodologia sistemática e de baixo custo para a proposição de modelos computacionais otimamente especificados em máquinas de vetores-suporte, os comitês de máquinas, particularmente os comitês de máquinas de vetores-suporte, se apresentam como alternativas promissoras. Basicamente, a abordagem de

comitês de máquinas agrega, de alguma forma, o conhecimento adquirido pelos componentes para chegar a uma solução global que é supostamente superior àquela obtida por qualquer um dos componentes isolados. A idéia de comitês de máquinas data de 1965, quando NILSSON (1965) considerou a estrutura de uma rede neural composta de uma camada de perceptrons elementares seguidos por um perceptron responsável por realizar um esquema de votação na segunda camada. Comitês de máquinas são aproximadores universais (HAYKIN, 1999) e podem se apresentar em versões estáticas, denominadas ensembles de componentes, ou então em versões dinâmicas, denominadas misturas de especialistas. Nesta tese, os componentes de um ensemble e os especialistas de uma mistura são tomados como máquinas de vetores-suporte. O pacote de software, contendo os algoritmos implementados e o manual de usuário, está disponível em:

http://www.lbic.fee.unicamp.br/machine_learning/package1

1.2 Objetivo da pesquisa

O objetivo fundamental da pesquisa é explorar conjuntamente as potencialidades advindas de máquinas de vetores-suporte e comitê de máquinas, adotando uma formulação unificada. Toda a abordagem está restrita ao caso de treinamento supervisionado. Várias extensões e novas configurações de comitês de máquinas de vetores-suporte são propostas, com análises comparativas que indicam ganho significativo de desempenho frente a outras propostas de aprendizado de máquina comumente adotadas para classificação e regressão.

Para atender tal objetivo, foram adotados os seguintes procedimentos:

- Apresentar o estado da arte em ensembles, misturas de especialistas e máquinas de vetores-suporte;
- Implementar uma extensão de SVM para ensembles de forma a agregar vantagens advindas de ambas as abordagens e procurar minimizar os riscos vinculados às especificações de projeto de máquinas de vetores-suporte. A abordagem proposta utiliza, dentro da mesma estrutura, diferentes máquinas de vetores-suporte, com funções de kernel distintas, e promove a configuração automática e sintonia da máquina de aprendizado, indicando automaticamente quais tipos de *kernel* são mais convenientes

para enfrentar com sucesso as peculiaridades da tarefa, incluindo conjuntos de dados ruidosos. Além disso, a combinação de vários mapeamentos de *kernel* com espaços de características diferentes, em um mesmo ensemble, tende a produzir melhor desempenho quando comparado com uma única máquina de vetores-suporte.

- Propor a incorporação de SVM em modelos de mistura de especialistas para formar uma mistura de SVMs (ME[SVM]). A formulação apresentada para ME[SVM] conduz a um problema quadrático similar àquele associado a uma única máquina de vetores-suporte, com manutenção da dimensão do problema de programação quadrática. Uma abordagem ME[SVM], além de permitir o uso de diferentes especialistas em regiões diferentes do espaço de entrada, também suporta a combinação de diferentes tipos de *kernel*.

1.3 Organização do texto

Os capítulos descrevem em detalhes todos os resultados obtidos na busca do atendimento dos objetivos genéricos enunciados na seção anterior. Sendo assim, iremos descrever a seguir a forma de organização em termos dos objetivos propostos. Todos os capítulos, com exceção dos capítulos 1 e 7, contêm um resumo introdutório.

Capítulo	Conteúdo
1	Inclui motivação, objetivos e organização do texto.
2	Apresenta um breve panorama da abordagem ensemble, mostrando as principais razões pelas quais um ensemble é capaz de apresentar um desempenho superior a qualquer componente tomado isoladamente. É também proposta uma taxonomia baseada nas principais formas que um componente-base pode ser gerado e combinado. Este capítulo será importante para fornecer subsídios ao Capítulo 5.

3	Expõe os principais conceitos de mistura de especialistas na forma de uma rede neural modular e hierárquica para aprendizado supervisionado. O modelo estatístico será discutido em detalhes, para motivar algumas possíveis extensões que serão propostas no Capítulo 6.
4	Expressa uma visão tutorial e abrangente acerca do formalismo e da aplicação de SVM junto a problemas de regressão e a problemas de classificação.
5	Explora conjuntamente algumas potencialidades advindas da abordagem SVM com aquelas da abordagem ensemble. Várias extensões e novas configurações de ensemble, tendo máquinas de vetores-suporte como componentes, são propostas, implementadas e aplicadas à solução de problemas de classificação e regressão, possibilitando assim a realização de análises comparativas de desempenho.
6	Realiza a incorporação de SVM ao formalismo de mistura de especialistas. Em virtude da existência de funções de *kernel* com melhor desempenho em determinadas regiões do espaço de entrada, a abordagem de mistura de especialistas se mostra promissora no sentido de indicar apropriadamente junto a qual região do espaço de entrada cada SVM deve se concentrar e qual deve ser a parametrização associada a cada especialista. Experimentos computacionais apontam para a validade da proposta e permitem posicionar misturas de SVMs dentre as mais eficazes ferramentas de aprendizado de máquina de propósito geral.
7	Contempla os comentários conclusivos e as perspectivas futuras.

Apêndice A	Descreve os conjuntos de dados utilizados para avaliar o desempenho das ferramentas propostas.
Apêndice B	Apresenta uma lista de publicações diretamente vinculadas à tese e que foram produzidas ao longo do plano de trabalho vinculado ao Projeto de Pesquisa.
Índice de Autores	Aponta as páginas do texto em que as referências bibliográficas foram citadas.
Referências	Contém a listagem das referências bibliográficas.

Capítulo 2

Ensembles

Resumo: Técnicas e conceitos de ensemble[1] representam uma das principais direções atuais em pesquisa na área de aprendizado de máquina, e têm sido aplicados a uma grande extensão de problemas práticos. Trata-se da obtenção de uma saída, para um problema de classificação ou de regressão, a partir das saídas individuais propostas por múltiplas soluções alternativas para o mesmo problema, denominadas componentes do ensemble. Apesar da falta de uma teoria de ensemble unificada, há muitas razões teóricas para combinar múltiplos componentes – classificadores ou regressores que irão compor o ensemble – e alguma evidência empírica da eficiência desta abordagem. Um ensemble consiste de um conjunto de componentes, cada um representando uma proposta de solução, cujas classificações, no caso de problemas de classificação de padrões, ou estimativas, no caso de problemas de regressão, são combinadas de formas diversas visando um ganho de desempenho. Pesquisas anteriores têm mostrado que um ensemble tende a ser mais preciso que qualquer um dos classificadores ou regressores que o compõem. *Bagging* (BREIMAN, 1996b) e *Boosting* (FREUND & SCHAPIRE, 1996; SCHAPIRE, 1990) são dois métodos populares e relativamente novos para a produção de ensembles.

Apesar de uma grande quantidade de publicações técnicas na área, não existe na literatura uma notação unificada para as formas de geração, seleção e combinação de

[1] A palavra ensemble será usada nesta tese conforme vem sendo realizado pela comunidade de aprendizado de máquina, em textos em língua portuguesa. Possíveis alternativas, como combinação ou agrupamento, não são capazes de exprimir apropriadamente a essência da metodologia, ou por representarem apenas uma das etapas envolvidas, ou por induzirem uma equivalência inexistente a outras metodologias modulares.

componentes. Neste capítulo, é apresentado um breve panorama da abordagem ensemble, mostrando as principais razões pelas quais um ensemble é capaz de apresentar um desempenho superior a qualquer componente. Será também proposta uma taxonomia baseada nas principais formas que um componente-base pode ser gerado e combinado. Além disso, no decorrer deste capítulo será adotada uma notação unificada para ensemble. Devido à grande variedade de propostas de implementação de ensembles, a revisão realizada cobre apenas os principais métodos empregados na literatura.

2.1 Introdução

Ensemble é um paradigma de aprendizado em que uma coleção finita de propostas alternativas para a solução de um dado problema, denominadas componentes do ensemble, é empregada em conjunto na proposição de uma única solução para o problema (S10, OLLICH & KROGH, 1996). Este paradigma originou-se do trabalho de HANSEN & SALAMON (1990), que mostraram que a habilidade de generalização pode ser significativamente melhorada por meio da composição de várias redes neurais artificiais, ou seja, treinamento independente de várias redes neurais artificiais e posterior composição das saídas individuais.

O bom desempenho dos ensembles tem incentivado o seu emprego em todas as linhas de atuação em aprendizado de máquina, de modo que outros componentes vêm sendo adotados para compor o ensemble, além de redes neurais artificiais (SHARKEY, 1999). Quanto às áreas de aplicação, pode-se mencionar:

- classificação de padrões: reconhecimento de face (GUTTA & WAIBEL, 1996; HUANG *et al.*, 2000), reconhecimento óptico de caracteres (DRUCKER, 1993; HANSEN *et al.*, 1992; MAO, 1998), análise de imagens (CHERKAUER, 1996), diagnóstico médico (CUNNINGHAM *et al.*, 2000; ZHOU *et al.*, 2000), classificação de sinais sísmicos (SHIMSHONI & INTRATOR, 1998).

- regressão: aproximação de funções na forma de mapeamentos de entrada-saída (HASHEM & SCHMEISER, 1993, 1995; LIMA *et al*., 2002), predição de séries temporais (WICHARD & OGORZALEK, 2004; INOUE & NARIHISA, 2000).

Em geral, um ensemble é construído em dois passos: geração dos vários componentes e então a combinação da saída proposta pelos componentes. Posteriormente, será mostrado que há uma tendência de ganho de desempenho quando se adota uma metodologia baseada em três passos: treinamento (geração de componentes), seleção e combinação. No entanto, para se adotar esta metodologia geralmente é necessário 3 conjuntos de dados, a saber, um conjunto de dados para geração dos componentes, um conjunto dados para seleção e combinação dos componentes e um conjunto para testar o desempenho do ensemble. Ambos conjuntos devem ser definidos de forma a manter a mesma distribuição amostral.

Quanto à geração dos componentes, as abordagens predominantes são *bagging* e *boosting*. *Bagging* (*B*ootstrap *agg*regat*ing*) foi proposto por BREIMAN (1996b), baseado em amostragem *bootstrap* (EFRON & TIBSHIRANI, 1993). Nesta abordagem, são gerados vários conjuntos de treinamento a partir de amostragem uniforme do conjunto original de dados, com reposição, e então se obtém uma proposta de solução a partir de cada um destes conjuntos de treinamento. Os conjuntos de treinamento têm o mesmo número de amostras do conjunto original, mas algumas amostras do conjunto original podem aparecer mais de uma vez, fazendo com que outras amostras não sejam selecionadas. Esta distinção aleatória entre os vários conjuntos de treinamento confere diversidade aos modelos de classificação ou regressão que são obtidos a partir de cada um desses conjuntos.

Boosting foi proposto por SHAPIRE (1990) e aperfeiçoado por FREUND & SCHAPIRE (1995) e FREUND (1995). Nesta abordagem, os vários conjuntos de treinamento não são gerados a partir de uma amostragem uniforme com reposição, como no caso do *bagging*. A probabilidade de escolha de uma amostra depende da contribuição desta para o erro de treinamento dos componentes já treinados, isto é, caso uma amostra não tenha sido corretamente classificada pelos componentes já gerados, a probabilidade de escolha desta aumenta em relação às demais amostras, quando do treinamento de novos componentes. Conseqüentemente, esta amostra terá uma chance maior de ser escolhida para compor o conjunto de dados do próximo componente a ser gerado. Portanto, apenas o primeiro

componente do ensemble é treinado a partir de uma amostragem uniforme do conjunto de dados original. É necessário, assim, que os vários componentes do ensemble sejam treinados seqüencialmente, visando redefinir a probabilidade de escolha das amostras na geração dos próximos conjuntos de treinamento.

Há muitas outras propostas para geração de componentes para um ensemble, além de *bagging* e *boosting*. Seguem alguns exemplos:

- HAMPSHIRE & WAIBEL (1990) utilizam critérios de erro distintos para treinar cada componente;
- CHERKAUER (1996) utiliza redes neurais artificiais multicamadas como componentes e adota cada componente com um número distinto de neurônios na camada intermediária;
- MACLIN & SHAVLIK (1995) também utilizam redes neurais artificiais multicamadas como componentes, mas as mantém com a mesma dimensão, sendo que a variação está na inicialização dos pesos sinápticos;
- OPITZ & SHAVLIK (1996) empregam um algoritmo genético para sintetizar componentes com comportamentos distintos, sendo que cada componente será aquele que obtiver melhor desempenho após a evolução completa de uma população de candidatos;
- YAO & LIU (1998) diferem de OPITZ & SHAVLIK (1996) no fato de que há apenas uma evolução completa, com os componentes do ensemble sendo tomados da população na última geração do algoritmo genético.

Quanto à combinação das redes neurais componentes, a abordagem mais predominante é o voto baseado na pluralidade ou voto majoritário (HANSEN & SALAMON, 1990), para problemas de classificação, e média simples (OPITZ & SHAVLIK, 1996) ou média ponderada (PERRONE & COOPER, 1993), para problemas de regressão.

Mas também há muitas outras abordagens para combinação de componentes. Seguem alguns exemplos:

- WOLPERT (1992) utiliza uma abordagem chamada empilhamento (do inglês *stacking*), em que os componentes são gerados via re-amostragem e combinados por um outro componente;

- MERZ & PAZZANI (1997) empregam regressão de componentes principais para determinar os pesos da combinação dos componentes e eliminar redundâncias (componentes com saídas similares);
- JIMENEZ (1998) calcula os pesos de ponderação da combinação, para um determinado padrão, de acordo com uma medida de confiança obtida a partir da saída dos componentes.

Observe que há algumas abordagens usando várias redes neurais para realizar uma tarefa no estilo dividir-e-conquistar (JACOBS *et al.*, 1991; JORDAN & JACOBS, 1994). Entretanto, nestas abordagens o componente é de fato treinado para diferentes sub-problemas ao invés do problema como um todo, o que faz com que estas abordagens usualmente sejam categorizadas como misturas de especialistas, ao invés de ensembles. Uma distinção mais detalhada é apresentada na próxima seção. No Capítulo 3, será apresentada uma visão geral de mistura de especialistas.

Embora a grande maioria destas propostas de ensemble considere todos os componentes disponíveis na combinação, a seleção de um subconjunto de componentes antes de se prosseguir à combinação tem se mostrado mais eficaz (ZHOU *et al.*, 2002), principalmente quando não se tem um controle acerca da qualidade absoluta de cada componente candidato e nem acerca de sua qualidade relativa aos demais candidatos. Como um exemplo, ZHOU *et al.* (2002) propuseram a abordagem GASEN para seleção de componentes, no caso, redes neurais artificiais, a partir de uma medida de aptidão associada a cada componente candidato.

A seguir, será realizada uma distinção entre as abordagens baseadas em ensemble e aquelas baseadas em módulos. Posteriormente, serão consideradas as principais motivações para utilizar ensembles e, em seguida, vários métodos de geração, seleção e combinação serão apresentados.

2.2 Distinção entre as abordagens ensemble e modular

Nesta tese, adota-se o termo comitê de máquinas (do inglês *committee machine*) para representar tanto abordagens ensemble quanto abordagens modulares, conforme a proposição de HAYKIN (1999). Esta perspectiva difere da adotada por BISHOP (1995), que associou o termo comitê apenas a ensembles. Faz-se necessário, portanto, estabelecer uma distinção entre a abordagem ensemble e a abordagem modular (SHARKEY, 1996).

O termo ensemble é aquele comumente empregado na combinação de um conjunto de componentes sintetizados para executar a mesma tarefa. Neste caso, cada componente representa isoladamente uma solução candidata para o problema de classificação ou regressão como um todo (HANSEM & SALAMON, 1990), podendo cada solução ser obtida por meios distintos e independentes entre si.

Já na abordagem modular, a tarefa é decomposta em várias sub-tarefas, e a solução da tarefa completa requer a contribuição de todos ou um subconjunto dos módulos componentes. Logo, um módulo isolado não representa uma solução candidata para o problema de classificação ou regressão como um todo, sendo fundamental a composição dos módulos. Dentre as abordagens modulares a serem estudadas nesta tese, destaca-se mistura de especialistas. Nesta abordagem, módulos chamados especialistas são combinados por meio de uma *gating*. No Capítulo 3, uma visão detalhada desta abordagem será apresentada.

2.3 Algumas razões para utilizar um ensemble

Diferentes propostas de solução podem explorar diferentes aspectos relevantes de um problema, enquanto muitas vezes uma única proposta de solução não é capaz de explorar todos os aspectos relevantes simultaneamente. Embora esta seja uma forte motivação para o emprego de um ensemble, existem outras motivações importantes a serem consideradas. Elas serão apresentadas vinculadas a problemas de classificação.

Considere o problema de classificação a partir de dados amostrados em sua formulação clássica. Dado um conjunto de exemplos de treinamento

$\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, onde $\mathbf{x}_i$, $i = 1,...,N$, são tipicamente vetores da forma $\mathbf{x}_i = [x_{i1} \quad x_{i2} \quad \cdots \quad x_{in}]^T$, cujos componentes são valores numéricos discretos ou contínuos (também chamados de características ou atributos de $\mathbf{x}_i$), e $y_i \in \{1,...,K\}$, $i = 1,...,N$, são índices que indicam a classe de $\mathbf{x}_i$. Será utilizada a notação x_{ij} para se referir à j-ésima característica de $\mathbf{x}_i$ e o número de classes K é tomado como sendo finito.

Considere um espaço de características $U = C_1 \cup C_2 \cup \ldots \cup C_K$ formado pela união de conjuntos mutuamente exclusivos, onde C_k é o conjunto de todas as possíveis amostras associadas à classe k, k=1,...,K. Um classificador f: $U \rightarrow \{1,...,K+1\}$ atribui a uma amostra $\mathbf{x} \in U$ um índice $k \in \{1,...,K+1\}$. Se $k \neq K+1$, a amostra é vinculada à classe k. Se $k = K+1$, então o classificador não foi capaz de associar a amostra $\mathbf{x}$ a uma das K classes. Na existência de um número finito M de propostas de classificadores, estas serão denotadas por $f_1, ..., f_M$.

Portanto, em um ensemble de classificadores, um número finito M de propostas de classificadores é combinado de alguma forma, tipicamente por voto majoritário (ponderado ou não ponderado), resultando em um único classificador. Uma das áreas mais ativas de pesquisa em aprendizado supervisionado é o estudo de métodos para construção de ensembles de classificadores capazes de produzir ganho de desempenho quando comparados ao desempenho de seus componentes isolados, independente das especificidades da aplicação.

Uma condição para que um ensemble de classificadores tenha capacidade de apresentar um desempenho superior a qualquer um de seus componentes individuais é fazer com que os classificadores que irão compor o ensemble tenham um bom desempenho isoladamente e sejam diversos entre si (HANSEM & SALAMON, 1990). Tomado isoladamente, um classificador seria considerado um candidato a compor um ensemble se apresentar, para todas as classes envolvidas, um desempenho superior àquele produzido por um classificador aleatório, ou seja, um classificador que rotula uma amostra qualquer de entrada com um índice aleatório que apresenta uma distribuição uniforme entre as K+1 classes candidatas. Dois classificadores são considerados diversos se eles não apresentam os mesmos erros de classificação frente a um mesmo conjunto de amostras, ou seja, se as

amostras classificadas erroneamente pelos classificadores diferem em algum grau. É evidente que podem ser definidos índices que meçam a qualidade individual e a diversidade entre os classificadores, indicando o grau de divergência.

Considere então um ensemble de M classificadores de bom desempenho, f_1, ..., f_M, adote o número de classes como sendo $K = 2$ (neste caso, não será considerada a classe K+1) e tome uma amostra arbitrária $\mathbf{x} \in U$. Suponha duas hipóteses:

- os erros de classificação apresentados pelos M classificadores são descorrelacionados;
- a probabilidade de classificar uma amostra arbitrária $\mathbf{x} \in U$ de forma equivocada é $p < 0{,}5$ para os M classificadores.

Logo, é alta a probabilidade da maioria deles classificar corretamente a amostra arbitrária $\mathbf{x} \in U$, com uma minoria realizando uma classificação equivocada. Nesta situação, um ensemble que combina os classificadores via voto majoritário vai produzir uma classificação correta. A questão que se apresenta agora é a seguinte: qual seria a probabilidade da maioria dos classificadores se equivocar ao classificar a amostra arbitrária $\mathbf{x} \in U$, prejudicando assim o desempenho do ensemble? Esta probabilidade é dada pela área da distribuição binomial para mais de $M/2$ classificações equivocadas (DIETTERICH, 2000). De fato, supondo $M = 21$ classificadores e a probabilidade de classificação equivocada de todos eles dada por $p = 0{,}3$, então a área da distribuição binomial para 11 ou mais classificadores equivocados é 0,026, a qual é muito menor que a probabilidade de erro dos classificadores individuais (DIETTERICH, 2000).

É evidente que, na prática, a probabilidade de classificação equivocada dos componentes do ensemble pode não ser muito baixa (embora todos a tenham como $p < 0{,}5$) e pode não haver descorrelação completa no erro de classificação entre os componentes, ainda mais quando se aumenta o número de componentes. Logo, o desempenho do ensemble sempre vai ser inferior ao máximo que se poderia obter teoricamente.

Em suma, pode-se conjecturar que o desempenho de um ensemble será melhor, tomando um ensemble arbitrário como referência, quando ocorrem simultaneamente as seguintes condições:

- o desempenho individual de cada componente é, na média, melhor que no ensemble arbitrário;

- o número de componentes é maior que no ensemble arbitrário;
- a descorrelação no erro de classificação entre os componentes é maior que no ensemble arbitrário.

Repare que as duas últimas condições acima são contraditórias, pois quanto maior o número de componentes menor será a descorrelação média entre eles, visto que todos apresentam probabilidade de erro de classificação inferior a 50%.

Esta caracterização formal do problema é relevante, mas não trata a questão da viabilidade prática da construção de bons ensembles para a tarefa de classificação. Felizmente, é comumente possível construir ensembles capazes de apresentar desempenho bem superior àquele apresentado pelos seus componentes, havendo três motivações fundamentais para isso (DIETTERICH, 2000). Que fique claro aqui que estas três motivações a serem apresentadas a seguir não se verificam sempre, o que inclusive ajuda a explicar por que um ensemble também pode apresentar desempenho inferior ao melhor de seus componentes tomado isoladamente.

A primeira motivação é estatística. A síntese do classificador pode ser vista como a busca em um espaço $\boldsymbol{H}$ de soluções candidatas ou hipóteses. O problema estatístico surge quando a quantidade de dados de treinamento disponíveis é limitada. Sem dados suficientes, o processo de síntese do classificador pode encontrar múltiplas propostas em $\boldsymbol{H}$, com todas elas apresentando o mesmo desempenho frente aos dados de treinamento. Se as condições para o emprego de ensembles se manifestarem em algum grau satisfatório, um ensemble baseado em voto majoritário pode reduzir o risco de classificação equivocada. A Figura 2.1(a) ilustra esta situação. A região mais ampla denota o espaço $\boldsymbol{H}$ de hipóteses. A região interna denota o conjunto de todos os classificadores que produzirão bom desempenho em relação ao conjunto de treinamento. O ponto rotulado f é o classificador desejado. Como as 4 hipóteses sugeridas estão distribuídas em torno de f, o ensemble tende a se aproximar de f mais do que qualquer uma das 4 hipóteses tomadas isoladamente.

A segunda motivação é computacional. Muitos algoritmos de busca iterativa em espaços de soluções candidatas trabalham realizando alguma forma de busca local gulosa, ou seja, que implementam localmente modificações junto à solução atual que sempre conduzem a melhorias incrementais de desempenho. Sendo assim, dependendo da condição

inicial adotada e/ou de decisões tomadas ao longo da execução da busca iterativa, há a possibilidade de convergência para um ótimo local, particularmente quando a multimodalidade (presença de múltiplos ótimos locais) é acentuada. Um ensemble construído pela execução de buscas locais iterativas a partir de vários pontos de partida diferentes no espaço de hipóteses ***H*** pode proporcionar uma melhor aproximação para *f*, como mostrado na Figura 2.1(b).

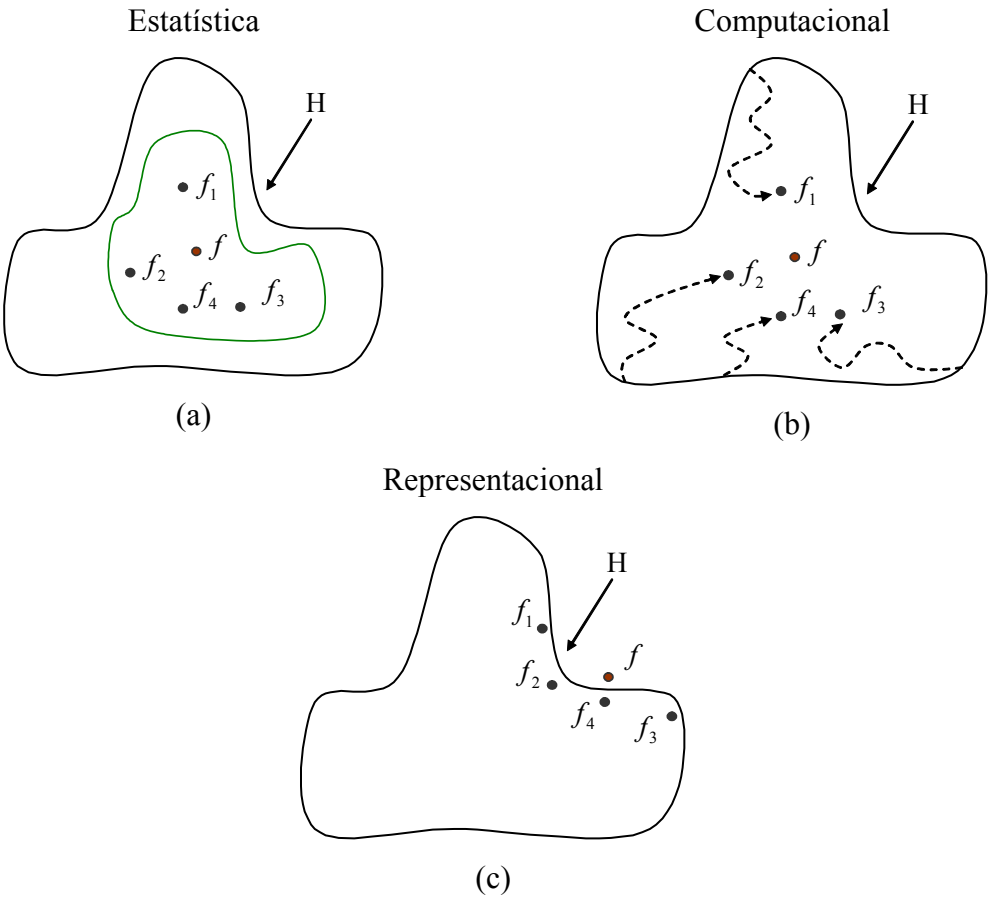


Figura 2.1 – Três motivações fundamentais para o emprego de um ensemble

A terceira motivação é representacional. Em muitas aplicações, o classificador ideal *f* não pode ser representado por nenhuma das hipóteses em ***H***. Tomando um ensemble de hipóteses representáveis em ***H***, pode ser possível estender o espaço de hipóteses representáveis de modo a se aproximar de *f*, conforme ilustrado na Figura 2.1(c). Mesmo

quando se adotam hipóteses de classificação associadas a representações matemáticas com capacidade teórica de aproximação universal, ou seja, com um espaço $\boldsymbol{H}$ ilimitado, na prática sempre vão existir limitações vinculadas à capacidade de representação computacional e de exploração global do espaço de busca.

A Figura 2.1 retrata, ao mesmo tempo, as fontes de degradação de desempenho para abordagens computacionais de síntese de classificadores, a partir de dados amostrados, e as fontes de motivação para o emprego de um ensemble.

Embora esta seção tenha se devotado ao estudo de problemas de classificação, conclusões equivalentes podem ser obtidas para o caso de problemas de regressão, onde a média é adotada no lugar do voto majoritário. Tendo examinado as principais motivações para a proposição de um ensemble, uma variedade de métodos para geração, seleção e combinação de componentes de um ensemble será apresentada ao longo das próximas três seções.

2.4 Geração de componentes de um ensemble

A principal motivação para a combinação de componentes em um ensemble é proporcionar incremento de desempenho, o qual é conquistado caso alguns requisitos sejam atendidos pelos componentes do ensemble. Sendo assim, a geração de componentes de um ensemble é uma etapa de grande relevância para sustentar a metodologia. Embora restrito ao caso de problemas de classificação, na seção 2.3 foram apresentados os principais requisitos a serem atendidos pelos componentes do ensemble: (1) cada um dos componentes deve apresentar bom desempenho quando tomado isoladamente; (2) o comportamento de cada componente deve ser o mais descorrelacionado possível frente aos demais.

Os resultados mais expressivos presentes na literatura, relativos à geração de componentes de ensemble, empregam redes neurais artificiais como componentes. Sendo assim, este também será o enfoque desta seção. O objetivo então é encontrar redes neurais de bom desempenho e que generalizam de forma descorrelacionada. Há vários parâmetros

de treinamento que podem ser manipulados tendo este objetivo em mente. Estes incluem: condição inicial, dados de treinamento, topologia da rede e algoritmo de treinamento.

No entanto, a simples adoção de parâmetros e aspectos estruturais distintos para as redes neurais componentes promove uma tendência, mas não garante a obtenção de padrões distintos de comportamento para as diversas soluções geradas, ou seja, não garante a obtenção de redes neurais de bom desempenho e que generalizam de forma descorrelacionada. Logo, os métodos mais avançados de geração de componentes devem ser dedicados à maximização da divergência de comportamento, ao mesmo tempo em que se busca garantir bom desempenho.

Considerando desde os métodos mais elementares até os mais avançados, eles podem ser divididos em duas classes: (1) geração pelo pré-processamento de parâmetros e aspectos estruturais de redes neurais artificiais; e (2) geração pelo pré-processamento dos dados de treinamento.

2.4.1 Pré-processamento de parâmetros e aspectos estruturais

As abordagens a serem apresentadas aqui exploram aspectos paramétricos e estruturais de cada candidato a compor o ensemble. Os aspectos a serem manipulados são considerados relevantes na definição da flexibilidade do componente em responder ao processo de treinamento supervisionado. Com diferentes flexibilidades, ou seja, com diferentes capacidades de resposta ao treinamento, os componentes tendem a generalizar de forma diversa. Embora sejam apresentados separadamente, é possível considerar mais de uma possibilidade de pré-processamento numa mesma implementação do processo de geração de componentes.

- **Definição do conjunto de pesos iniciais**: como os algoritmos de treinamento das redes neurais partem de um conjunto inicial de pesos, os quais são então submetidos a um processo iterativo de ajuste, a adoção de condições iniciais distintas representa pontos de partida distintos para o processo de ajuste de pesos, o que pode propiciar a convergência para um conjunto de pesos distinto, ou seja, a convergência do treinamento pode se dar para um ótimo local diferente e dependente da localização da

condição inicial ou ponto de partida. Com isso, espera-se que as redes neurais resultantes generalizem de forma diferente, mesmo que sejam mantidos idênticos os dados de treinamento, a arquitetura da rede neural e o algoritmo de treinamento.

- **Definição da arquitetura da rede neural**: como o comportamento de uma rede neural depende fortemente de sua arquitetura (basicamente vinculada à estratégia de conexão e ao número e tipo de neurônios), a adoção de arquiteturas distintas pode conduzir a redes neurais que generalizam de forma diferente, mesmo que sejam mantidos idênticos os dados de treinamento e o algoritmo de treinamento. Com arquiteturas distintas, os comportamentos de múltiplas redes neurais treinadas junto ao mesmo conjunto de dados podem ser bastante descorrelacionados, pois as estruturas internas de processamento serão distintas.
- **Definição do algoritmo de treinamento:** como os algoritmos de treinamento das redes neurais empregam um processo iterativo de ajuste, mesmo que se parta de uma mesma condição inicial, o processo de ajuste iterativo de pesos pode diferir bastante, de um algoritmo de treinamento para outro. Sendo assim, há a possibilidade de convergência para um conjunto de pesos distinto, ou seja, a convergência do treinamento pode se dar para um ótimo local diferente. Logo, a adoção de procedimentos de otimização distintos pode conduzir a redes neurais que generalizam de forma diferente, mesmo que sejam mantidos idênticos os dados de treinamento, os pesos iniciais e a arquitetura da rede neural.

2.4.2 Pré-processamento dos dados de treinamento

Os métodos que são mais freqüentemente empregados para a criação de ensembles são aqueles que envolvem pré-processamento dos dados de treinamento. Há várias formas diferentes de pré-processamento, podendo inclusive ser aplicadas simultaneamente em uma implementação computacional. Serão reservadas subseções específicas para detalhar as duas formas mais difundidas e que vêm apresentando um desempenho destacado.

O objetivo de todas as formas de pré-processamento é produzir conjuntos de treinamento distintos, que podem conduzir a redes neurais que generalizam de forma

diversa, mesmo que sejam mantidos idênticos os pesos iniciais, a arquitetura da rede neural e o algoritmo de treinamento.

- **Re-amostragem dos dados**: uma abordagem comum para a criação de um conjunto de componentes com capacidade de generalização distinta é a re-amostragem dos dados, caracterizada pela obtenção de vários sub-conjuntos de treinamento distintos a partir de um único conjunto de treinamento. Dentre os métodos de re-amostragem usados para este propósito está a técnica *bagging* (*b*ootstrap *ag*regat*ing*) (BREIMAN, 1996b), a qual será detalhada em uma próxima seção e envolve amostragem de dados com reposição.
- ***Boosting* ou re-amostragem adaptativa**: vários estudos empíricos (DRUCKER *et al.*, 1994) sustentam a eficácia do algoritmo *boosting*. FREUND & SCHAPIRE (1996) propuseram uma das mais utilizadas versões de *boosting*, chamada Adaboost (Ada*ptive* boos*ting*), em que os conjuntos de treinamento são re-amostrados de forma adaptativa, de tal modo que amostras que mais contribuem para o erro de treinamento dos componentes já treinados têm aumentada sua probabilidade de comporem o conjunto de treinamento a ser empregado na síntese do próximo componente. Como é deixado transparecer, os componentes do ensemble devem ser obtidos de maneira seqüencial. BREIMAN (1996a) explora algumas das diferenças entre os algoritmos Adaboost e *bagging*, concluindo, com base em evidência empírica e analítica, que o algoritmo Adaboost obtém melhor êxito que o *bagging* na redução da variância do erro de treinamento.
- **Conjuntos de treinamento disjuntos**: um método similar aos dois já mencionados acima envolve a obtenção de conjuntos de treinamento disjuntos ou mutuamente exclusivos, isto é, amostragem sem repetição (SHARKEY *et al.*, 1996). Não há então nenhuma sobreposição de dados usados para treinar os componente diferentes. Como observado por TUMER & GHOSH (1996), o problema é que o tamanho dos conjuntos de treinamento vai ficar reduzido, e este fator pode resultar em desempenho de generalização ruim.
- **Variáveis distintas e transformação de variáveis**: em algumas aplicações específicas (SHARKEY *et al.*, 1996), é possível obter conjuntos de treinamento distintos pela simples consideração de um elenco de variáveis distintas para o vetor de entrada. Desse modo,

cada componente do ensemble toma um subconjunto de variáveis de entrada distinto dos demais, embora possa haver variáveis comuns entre eles. Além da possibilidade de existirem variáveis provenientes de fontes sensoriais distintas na composição destes conjuntos, os diferentes conjuntos de variáveis podem ser gerados pela aplicação de métodos de poda de variáveis (TUMER & GHOSH, 1996), injeção de ruído (RAVIV & INTRATOR, 1999), ou então as variáveis distintas podem representar também alguma combinação linear ou não-linear de variáveis tomadas isoladamente em outros conjuntos (SHARKEY & SHARKEY, 1997), dentre as quais se destaca a análise de componentes principais lineares e não-lineares (SHARKEY & SHARKEY, 1997).

2.4.2.1 *Bagging* de componentes instáveis

O mais importante representante dentre as técnicas de re-amostragem de dados com reposição foi introduzido por BREIMAN (1996b), sob o nome de *bagging* (*b*ootstrap *ag*regat*ing*) ou agregação *bootstrap*. Trata-se de uma técnica de geração de conjuntos de treinamento distintos, os quais são utilizados para obtenção de componentes de um ensemble. A diferenciação do conjunto de treinamento tende a produzir componentes que generalizam de forma distinta.

O aspecto relevante da técnica está, portanto, na promoção de diversidade dos conjuntos de treinamento por amostragem *bootstrap* (EFRON & TIBSHIRANI, 1993). Uma vez dispondo de um único conjunto de treinamento com N amostras, chamado aqui de conjunto original, realiza-se a re-amostragem com reposição e mesma probabilidade de escolha de cada uma das N amostras, de modo a produzir M conjuntos de treinamento com N amostras cada. Logo, todas as amostras dos M conjuntos de treinamento gerados estão presentes no conjunto de treinamento original, de modo que a diferença entre os M conjuntos gerados está na presença de amostras repetidas e, por conseqüência, ausência de algumas amostras que compõem o conjunto original. Como há igual probabilidade de escolha de qualquer das N amostras e como N escolhas são realizadas para compor cada um dos M conjuntos de treinamento, então a reposição de amostras implica que uma amostra já escolhida possa ser escolhida novamente durante a composição de um dos conjuntos de

treinamento. Mais ainda, como a cardinalidade dos novos conjuntos se mantém em *N*, toda amostra repetida implica na ausência de alguma amostra do conjunto original, criando assim a diferença entre os *M* conjuntos de treinamento (todos com *N* amostras tomadas de um único conjunto original de *N* amostras).

Repare que, quanto maior o valor de *N*, maior é a probabilidade de ocorrência de amostras repetidas nos conjuntos de treinamento gerados, o que implica que a diferenciação entre os conjuntos gerados tende a aumentar proporcionalmente à sua cardinalidade *N*. Sendo assim, para valores suficientemente elevados de *N*, a re-amostragem com reposição representa uma técnica eficaz para a produção de conjuntos de treinamento representativos do conjunto de treinamento original e, ao mesmo tempo, distintos entre si.

No entanto, a existência de *M* conjuntos de dados distintos não implica que os *M* componentes a serem treinados irão generalizar de forma distinta. Para sustentar esta implicação, os componentes a serem treinados devem ser instáveis, no sentido de produzirem comportamento distinto sempre que submetidos a conjuntos de treinamento distintos. As redes neurais artificiais, assim como muitos outros modelos de regressão e classificação, são componentes instáveis (BREIMAN, 1996b). Isto pode ser verificado pelo fato de que a injeção de ruído de média zero junto aos dados de treinamento não implica em queda de desempenho na fase de treinamento, mas normalmente acaba produzindo uma degradação de desempenho em generalização. Fica claro então que a instabilidade não deve ser entendida sob a perspectiva de sistemas dinâmicos, mas sim sob a ótica da ausência de previsibilidade de comportamento de modelos de classificação e regressão após a fase de treinamento.

A teoria de regularização (TIKHONOV & ARSENIM, 1977) procura amenizar este efeito ao controlar o grau de flexibilidade do modelo de classificação ou regressão, reduzindo assim a instabilidade do mesmo. Esta redução de instabilidade é necessária, pois não é razoável que pequenas alterações no conjunto de treinamento levem a grandes modificações no comportamento do modelo de classificação ou regressão.

Bagging, por sua vez, ao buscar componentes que generalizam de forma distinta, não requer que os componentes (modelos de classificação ou regressão) generalizem otimamente, ou seja, não requer que a instabilidade seja evitada. Mesmo que os

componentes não apresentem uma boa capacidade de generalização, a agregação deles tende a generalizar bem.

Existe então um compromisso: a capacidade de generalização dos componentes não pode ser muito ruim, pois a melhora obtida com a agregação pode não ser suficiente, ao mesmo tempo em que a capacidade de generalização dos componentes não precisa ser muito boa, pois neste caso a agregação não produz ganho de desempenho. Logo, o grau ótimo de regularização dos componentes do *bagging* deve ser adequadamente sintonizado para cada problema de aplicação (TANIGUCHI & TRESP, 1997). Neste sentido, além da re-amostragem com reposição, alguns autores adotam mecanismos adicionais de perturbação dos conjuntos de treinamento gerados, como injeção de ruído (BREIMAN, 2000; DIETTERICH, 2000; RAVIV & INTRATOR, 1999; SHARKEY, 1999).

Já a agregação de componentes do *bagging* se dá de uma maneira muito simples:

- média simples das saídas dos M componentes, no caso de regressão;
- voto majoritário, no caso de classificação.

Estas e outras técnicas de agregação serão melhor apresentadas na seção 2.5, pois são também consideradas em diversas implementações de ensemble. Um pseudocódigo para o *bagging* é apresentado abaixo.

Sejam $\{P(1),\ldots,P(N)\}$ as probabilidades definidas para cada uma das N amostras de treinamento.

Tome $P(i) = 1/N$, $i=1,\ldots,N$.

Para $j = 1,\ldots,M$, faça:

    Gere um conjunto de treinamento contendo N amostras, utilizando as probabilidades $\{P(1),\ldots,P(N)\}$ e realizando amostragem com reposição;

    Realize o treinamento do j-ésimo componente a ser agregado.

Fim

A saída do ensemble, para o problema de classificação, é calculada usando voto majoritário e, para o problema de regressão, usando média simples.

2.4.2.2 *Boosting* para classificação

A principal diferença entre as abordagens *boosting* e *bagging* é que no *boosting* os componentes da agregação são treinados seqüencialmente, pois o conjunto de treinamento do próximo componente é gerado com base no desempenho dos demais componentes já

obtidos. Com isso, enquanto a abordagem *bagging* atua apenas na redução da variância, a abordagem *boosting* é capaz de reduzir tanto variância quanto bias (FRIEDMAN *et al.*, 2000). Isto se deve à atribuição de maior ênfase às amostras responsáveis pela queda de desempenho durante o treinamento, ou seja, as amostras que mais contribuem para o erro de treinamento dos componentes já treinado têm maior probabilidade de serem escolhidas na composição do conjunto de treinamento do próximo componente. Lembre-se que, na abordagem *bagging*, a probabilidade de escolha das amostras é sempre uniforme, razão pela qual o treinamento dos componentes pode ser feito em paralelo, ou seja, simultaneamente.

A abordagem *boosting* pode ser implementada de três formas:

- *Boosting* por filtragem;
- *Boosting* adaptativo (AdaBoost);
- *Boosting* por re-ponderação.

A abordagem original de *boosting*, denominada *boosting por filtragem*, é devida a SCHAPIRE (1990), Aqui, três componentes são usados e esta técnica supõe a disponibilidade de uma quantidade muito grande de amostras candidatas a compor o conjunto de treinamento. O primeiro componente é treinado tomando-se N_1 amostras de treinamento, obtidas de um conjunto de N amostras, sendo que a definição do valor de N_1 depende da aplicação. Então o segundo componente é treinado com base em um conjunto de treinamento contendo N_2 amostras. Estes dados de treinamento são selecionados a partir dos dados de treinamento contendo N amostras, tal que 50% destas amostras foram classificadas corretamente pelo primeiro componente. O terceiro componente, por sua vez, é treinado com N_3 amostras, as quais são caracterizadas pelo fato de que os dois componentes já treinados discordam quanto à classe a ser atribuída a elas. A classificação é então realizada pelo voto majoritário dos três componentes.

A motivação original para *boosting* veio da teoria de aprendizado PAC (do inglês PAC *learning – probably approximately correct learning*) (VALIANT, 1984). A técnica requer que os componentes do agrupamento, no caso, redes neurais artificiais, apresentem pelo menos um desempenho superior àquele obtido por um classificador que atribui a classe de cada amostra de forma aleatória.

Boosting adaptativo ou AdaBoost é uma combinação de idéias relacionadas a *boosting* por filtragem e *bagging*, não requerendo que o conjunto original de treinamento seja muito grande (FREUND & SCHAPIRE, 1996). Muitas variações do AdaBoost foram sugeridas na literatura. Será apresentado a seguir o algoritmo na forma proposta por BREIMAN (1999).

Como no caso da abordagem *bagging*, membros do ensemble são treinados usando re-amostragem com reposição, mas aqui a probabilidade de seleção de uma amostra depende do desempenho dos classificadores treinados anteriormente. Se uma amostra *i* foi incorretamente classificada pelo classificador *j* ($a(i) = 1$), aumentará a probabilidade dela estar presente no conjunto de treinamento do classificador *j*+1. E este incremento de probabilidade será ainda maior caso o desempenho geral do classificador *j* for bom (indicado por um elevado valor de β_j). Finalmente, os pesos do *j*-ésimo classificador na votação do agrupamento é $\log(\beta_j)$, atribuindo mais ênfase aos classificadores que se desempenham bem.

Sejam $\{P(1),\ldots,P(N)\}$ as probabilidades definidas para cada uma das *N* amostras de treinamento.

Tome inicialmente $P(i) = 1/N$, $i=1,...,N$.

Para $j = 1,\ldots,M$, faça:

Gere um conjunto de treinamento contendo *N* amostras, utilizando as probabilidades $\{P(1),\ldots,P(N)\}$ e realizando amostragem com reposição;

Realize o treinamento do *j*-ésimo componente a ser agregado.

Para $i=1,...,N$, tome $\begin{cases} a(i) = 1 & \text{se o } i\text{-ésimo padrão foi classificado corretamente;} \\ a(i) = 0 & \text{caso contrário.} \end{cases}$

Defina:

$$\varepsilon_j = \sum_{i=1}^{N} a(i)P(i) \text{ e } \beta_j = (1-\varepsilon_j)/\varepsilon_j .$$

Atualize a probabilidade de cada uma das *N* amostras da seguinte forma: $P(i) = c_j P(i)\beta_j^{d(i)}$, $i = 1,...,N$, onde c_j é o fator que garante que a soma das probabilidades permanece unitária.

Fim

A saída do ensemble é calculada usando voto majoritário $Classe(\mathbf{x}) = \arg\max_k \sum_{j=1}^{M} \log(\beta_j) f_{j,classe=k}(\mathbf{x})$, onde *k* é o índice da classe, $\log(\beta_j)$ são os pesos de ponderação e $f_{j,classe=k}(\mathbf{x})$ é 1 se o classificador *j* classifica a amostra **x** como sendo da classe *k*, e 0 caso contrário.

Outra variação do AdaBoost é o *boosting por re-ponderação*. Nesta versão determinística, as probabilidades $\{P(1),\ldots,P(N)\}$ não são usadas para re-amostragem, mas apenas para ponderar os erros cometidos pelos componentes para aquelas amostras, definindo assim o quão relevante é errar ou acertar a classe de uma amostra específica. Por exemplo, suponhamos que uma amostra não seja classificada corretamente pelos componentes obtidos anteriormente; logo sua probabilidade em relação aos demais será muito alta; conseqüentemente, o próximo componente tentará acertar a classe desta amostra, caso contrário o erro alcançado por este será muito alto. Em FRIEDMAN *et al.* (2000), algumas variações destas abordagens determinísticas são descritas.

Como uma extensão, AdaBoost pode ser usado na obtenção de classificadores que produzem a probabilidade das classes, abrindo assim perspectivas para sua aplicação junto a problemas de regressão (ZEMEL & PITASSI, 2001). Pesquisas recente mostraram que há conexão entre AdaBoost e classificadores que operam com a maximização da margem de separação (do inglês *large margin classifiers*), incluindo máquinas de vetores-suporte (SCHAPIRE *et al.*, 1998; DRUCKER, 1999).

Já para amostras com altas taxas de ruído, RATSCH e colaboradores propuseram o AdaBoost regularizado (RÄTSCH *et al.*, 1999) e o *v*-Arc (RÄTSCH *et al.*, 2000). Ambos são algoritmos *boosting* que toleram *outliers* na classificação e apresentam bom desempenho para casos com sobreposição de classes.

2.4.2.3 *Boosting* para regressão

FREUND & SCHAPIRE (1996) propuseram uma extensão do Adaboost para problemas de regressão, chamada Adaboost.R. O algoritmo a seguir é uma modificação do Adaboost.R proposta por DRUCKER (1997). A vantagem desta modificação é permitir outros tipos de função de perda.

Sejam $\{P(1),\ldots,P(N)\}$ as probabilidades definidas para cada uma das N amostras de treinamento.

Tome inicialmente $P(i) = 1/N$, $i = 1,...,N$.

Para $j = 1,\ldots,M$, faça:

Gere um conjunto de treinamento contendo N amostras, utilizando as probabilidades $\{P(1),\ldots,P(N)\}$ e realizando amostragem com reposição;

Realize o treinamento do j-ésimo componente a ser agregado.

Calcule a saída do j-ésimo componente para todo o conjunto de treinamento $f_j(\mathbf{x}_i)$, $i = 1,\ldots,N$.

Calcule uma perda para cada exemplo de treinamento conforme $L_i = L\left[\left|f_j(\mathbf{x}_i) - y_i\right|\right]$. A perda L pode ter qualquer funcional, desde que $L \in [0,1]$. Se for utilizado, por exemplo,

$$D = \sup\left|f_j(\mathbf{x}_i) - y_i\right| \quad i = 1,\ldots,N$$

então foram propostas três funções de perda possíveis:

$$L_i = \frac{\left|f_j(\mathbf{x}_i) - y_i\right|}{D} \text{ (Linear); } L_i = \frac{\left|f_j(\mathbf{x}_i) - y_i\right|^2}{D^2} \text{ (quadrática); } L_i = 1 - \exp\left\{-\frac{\left|f_j(\mathbf{x}_i) - y_i\right|}{D}\right\}$$

(exponencial)

Calcule uma perda média: $\overline{L} = \sum_{i=1}^{N} L_i P(i)$

Calcule $\beta = \frac{\overline{L}}{1-\overline{L}}$

Atualize a probabilidade de cada uma das N amostras na forma: $w_i \rightarrow w_i \beta \exp\{1 - L_i\}$, i=1,...,N.

Se $\overline{L} \leq 0.5$, repita o processo, caso contrário finalize.

Fim

Obtenha a predição f usando os M preditores, sendo que, para uma entrada particular $\mathbf{x}_i$, cada um dos M componentes deve fazer uma predição f_j, $j = 1,\ldots,M$, resultando:

$$f = \inf\left\{y \in Y : \sum_{j:f_j \leq y} \log(1/\beta_j) \geq \frac{1}{2}\sum_{j=1}^{M} \log(1/\beta_j)\right\}.$$

Este valor representa a mediana ponderada, onde cada componente f_j faz uma predição f_j^i sobre o i-ésimo padrão e um β_j associado. Para um padrão i, as predições são re-rotuladas de forma que (mantendo a associação β_j com f_j^i):

$$f_1^i < f_2^i <,\ldots,< f_M^i$$

Para obter o valor f, soma-se $\log(1/\beta_j)$ até encontrar o menor j tal que a desigualdade acima seja satisfeita. A predição daquele componente j é, então, tomada como a predição do ensemble. Caso os β_j fossem todos iguais, estes deveriam reproduzir a própria mediana.

A variação dos pesos implica em atribuir aos componentes subseqüentes um conjunto de treinamento com exemplos mais difíceis para treinar. Desta forma, a perda média tende a aumentar quando o algoritmo é repetido, não satisfazendo o limiar $\overline{L}$, o que leva ao encerramento do algoritmo.

Embora o algoritmo acima seja similar ao Adaboost.R, não há nenhuma garantia de que ele irá terminar em um número finito de passos, considerando um erro de treinamento zero (DRUCKER, 1997). Considerando erro de treinamento zero, a convergência depende da obtenção de um algoritmo de aprendizado de máquina tal que $\overline{L} < 0.5$. Quando se trata de dados reais com máquina de aprendizado real, o número de exemplos difíceis no conjunto de treinamento aumenta a cada iteração, tornando-se mais difícil encontrar tal limite. Embora seja possível encontrar uma máquina de aprendizado real que sempre encontrasse tal limite, o algoritmo *boosting* não diz nada a respeito do desempenho sobre um conjunto de teste. Portanto, é necessário assegurar que, ao decrementar o erro de treinamento, não haverá sobre-ajuste às peculiaridades de um conjunto de treinamento e que haverá boa generalização sobre os dados de teste. A forma de fazer isto dependente da implementação específica da máquina de aprendizado.

2.5 Métodos de combinação para um ensemble

Métodos de combinação surgiram na Estatística (FRENCH, 1985) e em Pesquisa Operacional (BATES & GRANGER, 1969). Muitos métodos combinam não somente o rótulo das classes, fornecido por cada componente, mas também as saídas dos componentes. Uma vez que um conjunto de componentes foi criado, uma forma efetiva de combinação de algumas de suas saídas deve ser encontrada. Há vários métodos diferentes de combinação, e dado que existem várias revisões sobre o assunto (JACOBS, 1995; GENEST & ZIDEK, 1996; XU *et al.*, 1992; ZHILKIN & SOMORJAI, 1996), nesta seção serão apresentados apenas alguns métodos clássicos de combinação de componentes de um ensemble. Como os métodos de combinação são específicos para problemas de classificação e regressão, haverá seções distintas considerando cada caso.

2.5.1 Métodos de combinação de múltiplos classificadores

Baseado na definição das seções anteriores sobre a decisão de um classificador, o problema pode ser descrito da seguinte maneira: trata-se da combinação de M classificadores diferentes f_j, $j = 1, \ldots, M$, utilizados para classificação do padrão $\mathbf{x}$ em K classes, $U = \{1, \ldots, K\}$. A saída do j-ésimo classificador para a k-ésima classe será denotada por μ_j^k. A saída do ensemble será denotada por $\hat{f}(\cdot)$. Logo, a combinação de múltiplos componentes pode ser vista como a determinação de $\hat{f}(\mathbf{x})$ usando as saídas $f_j(\mathbf{x})$ de todos os M componentes.

Média e média ponderada: é um dos métodos mais populares de combinação. Ele se refere a uma combinação linear da saída dos componentes. Uma única saída pode ser criada a partir de um conjunto de saídas via média simples (PERRONE & COOPER, 1993), ou por meio de uma média ponderada, a qual considera o desempenho relativo dos componentes a serem combinados (PERRONE & COOPER, 1993; HASHEM, 1993; HASHEM, 1996; HASHEM, 1997). Neste método, a saída de um ensemble será a classe com maior valor médio, conforme mostra a equação a seguir:

$$\hat{f}(\mathbf{x}) = \max_{k \in U} S_k, \tag{2.1}$$

onde $S_k = \frac{1}{M} \sum_{j=1}^{M} \mu_j^k(\mathbf{x})$.

Métodos de combinação não-linear: dentre os diversos métodos de combinação não-linear que têm sido propostos, existem aqueles que se baseiam na crença de Dempster-Shafer (ROGOVA, 1994), na combinação por meio de rank (AL-GHONEIM & KUMAR, 1995), na combinação por votação (HANSEM & SALAMON, 1990) e na combinação por regras estatísticas (TUMER & GHOSH, 1995).

Generalização empilhada: proposta por WOLPERT (1992), um componente (por exemplo, uma rede neural) aprende a combinar as saídas dos componentes. A saída do conjunto de

componentes (chamados generalizadores de nível 0) é usada como a entrada para o componente responsável pela combinação (chamado generalizador do nível 1), que é responsável por produzir a saída apropriada. Um estudo aprofundado sobre empilhamento é apresentado em LEBLANC & TIBSHIRANI (1993).

Votação: O resultado apoiado pela maioria dos componentes é definido como a saída de um ensemble, segundo a equação a seguir:

$$\hat{f}(\mathbf{x}) = k^* \tag{2.2}$$

se $k^* = \arg\max_k \left(\sum_{j=1}^{M} G_j^k(\mathbf{x}) \right)$, onde

$$G_j^k(\mathbf{x}) = \begin{cases} 1 \text{ se } f_j(\mathbf{x}) = k \\ 0 \text{ se } f_j(\mathbf{x}) \neq k \end{cases}. \tag{2.3}$$

Embora a votação tenha a vantagem de não requerer recursos computacionais ou espaço de memória extra, esta pode piorar o desempenho do ensemble, caso existam componentes com desempenho global ruim. Isto ocorre porque os rótulos de todos os componentes têm o mesmo peso, independente do seu desempenho global. O princípio de votação é exatamente o que nós conhecemos como votação majoritária. Algumas variações desta idéia foram propostas para problemas de múltiplas classes, como segue.

Unanimidade: O ensemble decide que uma observação **x** pertence à classe *k* se e somente se todos os componentes tomarem a mesma decisão.

Unanimidade modificada: O ensemble decide que uma observação **x** pertence à classe *k* se quase todos os componentes concordarem com a observação. A margem de discordância admissível é parâmetro do algoritmo.

Majoritário (ponderado ou não-ponderado): Neste caso, o ensemble decide que uma observação **x** pertence à classe *k* se mais que a metade (ou eventualmente outra proporção

em relação ao número de componentes) dos componentes concordarem com esta decisão. É evidente que este e o item anterior diferem apenas na margem de discordância aceitável.

Pluralidade Limitada: O ensemble decide que a observação **x** pertence à classe *k* se o número de componentes que a classificam desta maneira é maior que o número de componentes que a classificam em qualquer outra classe.

A combinação de classificadores por meio destes métodos é simples, não requerendo qualquer conhecimento anterior do comportamento do classificador, nem qualquer metodologia complexa para tomar a decisão. É necessária apenas a contagem do número de classificadores que concordam e discordam de uma decisão para se definir a classe para o padrão de entrada atual. Entretanto, há um problema: os pesos da decisão de todos os classificadores são iguais, independentemente do desempenho geral de cada classificador. Dentre os diversos métodos que levam em consideração o desempenho dos componentes na tomada de decisão, tem-se os seguintes: Métodos Bayesianos (FRENCH, 1985; LEE, 1997), Supra-Bayesianos (JACOBS, 1995), Método BKS (do inglês *Behavior-Knowledge Space*) (HUANG & SUEN, 1995; WERNECKE, 1992), Método Dempster-Shafer (XU *et al.*, 1992), Regressão Logística (HO *et al.*, 1994).

Método Ideal: Este método não representa uma opção factível, pois o seu emprego requer o conhecimento da saída desejada. Aqui, sempre é escolhida a saída correta caso haja algum componente que a produz. Portanto, desde que haja pelo menos um componente que produz uma saída correta, o ensemble vai produzir uma saída correta para a observação **x**. A utilidade deste método está na produção de medidas relativas de desempenho para os outros métodos de combinação, visto que ele não pode ser superado por nenhum outro método de combinação linear.

2.5.2 Métodos de combinação de múltiplos regressores

Nesta seção, serão apresentados os métodos básicos de combinação utilizados na literatura, propostos por HASHEM (1997). Considere o problema de aproximação de um mapeamento com *n* entradas e uma saída, isto é, $f : \Re^n \rightarrow \Re$. Uma abordagem para o caso de *d* saídas é tratá-las separadamente. Tal tratamento independente é direto (HASHEM *et al.*, 1993) e minimiza o EQM total para o mapeamento de várias entradas e várias saídas.

O problema de combinação pode ser descrito da seguinte maneira: trata-se da combinação de *M* estimadores diferentes f_j, $j = 1,\ldots,M$, utilizados para estimação do padrão **x**. O *j*-ésimo componente recebe como entrada um vetor **x** e retorna uma saída escalar $f_j(\mathbf{x})$. O erro de aproximação é dado por $\delta(\mathbf{x}) = y(\mathbf{x}) - f_j(\mathbf{x})$, onde $y(\mathbf{x})$ é a resposta desejada considerando a entrada **x.** A saída do ensemble será denotada por $\hat{f}(\cdot)$. Logo, a combinação de múltiplos estimadores pode ser vista como a determinação de $\hat{f}(\mathbf{x})$ usando as saídas $f_j(\mathbf{x})$ de todos os *M* componentes.

2.5.2.1 Média simples

É um dos métodos de combinação mais populares. Consiste em realizar uma combinação linear sobre a saída de *M* componentes, dada por:

$$\hat{f}(\mathbf{x}) = \frac{1}{M}\sum_{j=1}^{M} f_j(\mathbf{x}). \qquad (2.4)$$

Uma desvantagem deste critério é o fato de considerar que as saída dos *M* componentes são igualmente importantes.

2.5.2.2 Média ponderada

De acordo com HASHEM & SCHMEISER (1995), uma combinação ponderada das saídas de *M* componentes retorna uma saída escalar $\hat{f}(\mathbf{x},\boldsymbol{\alpha}) = \sum_{j=1}^{M} \alpha_j f_j(\mathbf{x})$ com erro de aproximação $\delta(\mathbf{x}) = y(\mathbf{x}) - \hat{f}(\mathbf{x},\boldsymbol{\alpha})$, onde $f_j(\mathbf{x})$ é a saída do *j*-ésimo componente e α_j (*j* =

1,...,*M*) são os pesos da combinação linear. HASHEM *et al.* (1994) estenderam esta definição de $\hat{f}(\mathbf{x},\boldsymbol{\alpha})$ para incluir um termo constante $\alpha_0 f_0(\mathbf{x})$, onde $f_0(\mathbf{x})=1$. Este termo constante permite a correção de qualquer polarização em $f_j(\mathbf{x}), j=1,...,M$. Assim $\hat{f}(\mathbf{x},\boldsymbol{\alpha})$ é dada por:

$$\hat{f}(\mathbf{x},\alpha)=\sum_{j=0}^{M}\alpha_j f_j(\mathbf{x})=\boldsymbol{\alpha}^T f(\mathbf{x}), \tag{2.5}$$

onde $\boldsymbol{\alpha}$ e $f(\mathbf{x})$ são vetores $(M+1)\times 1$. Dessa forma, o problema é encontrar valores apropriados para os pesos da combinação α_0, α_1, ..., α_M. Uma abordagem possível é encontrar o melhor componente dentre os *M* componentes e fixar o seu respectivo peso em 1 e os demais em 0. Usando o melhor componente, tem-se a vantagem da simplicidade, porém existe o risco de perder informações úteis presentes nos demais $M-1$ componentes. Outra abordagem largamente difundida é usar pesos iguais para a combinação. Esta abordagem assume que todos os componentes contribuem igualmente, conforme já mencionado anteriormente.

A combinação linear ótima (OLC) (do inglês *optimal linear combination*), proposta por HASHEM & SCHMEISER (1995), consiste em encontrar o vetor de pesos da combinação $\boldsymbol{\alpha}=[\alpha_0 \quad \alpha_1 \quad \cdots \quad \alpha_M]^T$ que minimiza o erro quadrático médio (EQM):

$$\min_{\boldsymbol{\alpha}} EQM\left[\hat{f}(\mathbf{x},\boldsymbol{\alpha})\right]. \tag{2.6}$$

HASHEM & SCHMEISER (1995) propuseram quatro formas diferentes para encontrar os pesos da combinação. O que diferencia tais propostas é a inclusão ou não de um termo constante (bias) e a restrição ou não para que os pesos da combinação tenham soma unitária.

2.6 Implementação de ensembles efetivos

Nas seções anteriores, foi apresentada uma descrição das diferentes possibilidades de geração e combinação de componentes de um ensemble. A questão considerada agora é como criar um ensemble efetivo, ou seja, um ensemble que apresente um desempenho melhor que o melhor componente tomado isoladamente. Independente do tipo de combinação adotada, não há garantia de ganho de desempenho frente ao melhor

componente isolado, pois a eficiência de um ensemble depende da forma como seus componentes se comportam em relação aos erros de predição (ROGOVA, 1994). Uma vez que a importância da descorrelação do erro entre os componentes já foi reconhecida, existem três perspectivas de abordagem a serem adotadas:

- considerar a dependência entre os componentes quando se escolhe um método de combinação;
- criar componentes objetivando uma combinação efetiva;
- selecionar componentes para uma combinação efetiva.

2.6.1 Consideração da dependência entre os componentes

Métodos de combinação que levam em consideração a dependência entre os componentes já foram propostos na literatura. Na seção 2.5, já foram apresentados alguns métodos de combinação que levam em consideração o desempenho dos componentes, em contrapartida ao uso de pesos iguais pela combinação via média simples.

O grau de descorrelação entre componentes fornece uma boa medida de como estes deveriam ser combinados. Por exemplo: considerando um problema de classificação, se os componentes não exibem nenhum erro de classificação coincidente com relação a um conjunto de validação, então a combinação desses componentes por meio de voto majoritário tende a produzir bons resultados. Um bom desempenho também tende a ser obtido quando o voto majoritário é usado para combinar componentes que compartilham erros, desde que o compartilhamento de erros não envolva a maioria dos componentes, junto a cada padrão do conjunto de validação.

Quando estas condições são violadas, ainda assim é possível obter bom desempenho, mas será necessário empregar métodos de combinação mais sofisticados, tais como generalização empilhada (WOLPERT, 1992) ou alguma forma de média ponderada (HASHEM, 1993; 1996; 1997; PERRONE & COOPER, 1993).

2.6.2 Geração de componentes buscando a efetividade do ensemble

Vários pesquisadores têm investigado empiricamente a eficiência dos diferentes métodos de criação de componentes. Há uma tendência na literatura em se apontar o pré-processamento dos dados como mais efetivo que o pré-processamento de parâmetros e aspectos estruturais (PARMANTO *et al.*, 1996; SHARKEY *et al.*, 1996; TUMER & GHOSH, 1996). A efetividade aqui implica uma maior probabilidade de gerar componentes que generalizam de forma diversa.

É evidente a dificuldade em argumentar conclusivamente contra a efetividade da alteração das condições iniciais de rede neurais artificiais que irão compor um ensemble, pois não se pode considerar como resolvidas as questões de geração de condições iniciais efetivas e de algoritmos de treinamento capazes de explorar convenientemente esta diversidade.

ROSEN (1996) propõe uma abordagem para pré-processamento dos dados em que as redes neurais são forçadas a serem descorrelacionadas entre si, por meio de um algoritmo de treinamento que incorpora um termo de penalidade para a correlação dos componentes. Desta forma, busca-se gerar seqüencialmente componentes que tendem a se comportar de forma descorrelacionada.

2.6.3 Métodos avançados para seleção de componentes

Há duas questões que podem ser consideradas em relação à seleção:

(i) o porquê dela ser empregada; e

(ii) como realizá-la.

Antes de descrever técnicas avançadas de seleção de componentes para ensembles, é importante apresentar a seleção como uma forma de melhorar o desempenho do ensemble. Visto que nem todos os componentes gerados irão compor o ensemble, eles serão denotados aqui de candidatos a compor o ensemble. O objetivo da seleção é maximizar o desempenho de generalização do ensemble pela definição de um subconjunto dentre o total de candidatos a compor o mesmo.

Os trabalhos de HASHEM (1996) e PERRONE & COOPER (1993) argumentam que a presença de colinearidade reduz a eficiência do ensemble. Embora o pré-processamento dos dados possa produzir, de forma efetiva, componentes que generalizam diferentemente, uma exceção ocorre quando a noção de conjunto de treinamento representativo torna-se aparente. O argumento é que conjuntos de treinamento disjuntos não necessariamente resultam em baixa correlação dos componentes resultantes do processo de treinamento. Este ponto pode ser explicado referenciando o conceito de conjunto de treinamento representativo (DENKER *et al.*, 1987; SHARKEY & SHARKEY, 1995). Conjuntos de treinamento representativos são aqueles que possibilitam que diferentes ferramentas computacionais que apresentem capacidade de aproximação universal convirjam para comportamentos altamente correlacionados, mesmo que cada uma delas trabalhe com conjuntos de treinamento diferentes (mas gerados pelo mesmo processo amostral).

Por outro lado, se um conjunto de componentes candidatos fosse treinado usando conjuntos de treinamento não-representativos, cada uma dos componentes deveria apresentar comportamentos com um maior grau de descorrelação e, assim, mostrar padrões de generalização diferentes para o conjunto de teste. Contudo, o desempenho individual de cada componente pode ser muito ruim. Logo, embora o ensemble possa apresentar ganhos de desempenho acentuados frente ao melhor componente isolado, o seu desempenho pode ainda assim ser ruim, conforme observado por TUMER & GHOSH (1996).

Há, portanto, um delicado balanço entre a representatividade do conjunto de treinamento e a correlação do comportamento apresentado pelos candidatos a compor o ensemble. Como conseqüência, somente um processo de teste de desempenho de cada proposta de ensemble é capaz de indicar se aquela combinação vai ou não ser efetiva.

Esta mesma noção de conjunto de treinamento representativo foi utilizada por BREIMAN (1999) para sugerir que os candidatos treinados para compor o ensemble deveriam ser sub-regularizados, uma vez que a variância resultante fosse removida pela combinação. Mas, a discussão assumiria outras perspectivas quando se considera a existência de níveis elevados de ruído junto às amostras de treinamento. Assim, RAVIV & INTRATOR (1999) obtiveram melhores resultados quando incorporaram uma forma de regularização (decaimento dos pesos) no treinamento de seus componentes (redes neurais)

sobre dados ruidosos. Uma discussão interessante do efeito do sobre-ajuste na combinação de candidatos a compor um ensemble pode ser encontrada em SOLLICH & KROGH (1996).

A questão agora é: como selecionar? Uma abordagem considera a criação de um reservatório de candidatos a compor o ensemble e um critério de seleção para detectar o melhor ensemble dentre todas as possibilidades de seleção de candidatos e de técnicas de combinação. Esta é uma abordagem adotada por PERRONE & COOPER (1993). Eles sugerem um método de seleção pelo qual os componentes treinados de população são ordenados segundo o erro quadrático médio crescente. De posse desta lista ordenada, o ensemble pode ser formado apenas pelos candidatos de melhor desempenho, até um determinado valor de corte, ou então uma técnica construtiva pode ser adotada, com a inclusão de componentes que conduzam a um incremento na capacidade de generalização junto a um conjunto de dados de seleção. Este conjunto de dados deve ser escolhido de forma a manter a mesma distribuição amostral dos dados de treinamento. Caso isto não aconteça, o ensemble gerado poderá apresentar desempenho ruim quando for apresentado o conjunto de teste.

A manipulação de um reservatório de candidatos pode se dar de duas formas:

(i) aplicar procedimentos de seleção a um conjunto de candidatos que foram gerados por meio do uso de métodos concebidos para promover diversidade; ou

(ii) realizar um processo contínuo de geração e seleção até que um critério de parada seja alcançado.

A segunda destas duas possibilidades é explorada por OPITZ & SHAVLIK (1996), que apresenta um método que usa algoritmos genéticos para encontrar componentes para ensemble que generalizem bem e que apresentem baixa correlação. Trata-se de um processo evolutivo que trabalha com uma população de candidatos. Os candidatos selecionados para compor o ensemble são combinados usando a média ponderada.

O critério de parada de um processo de seleção de componentes deve depender dos recursos computacionais disponíveis e do nível de desempenho esperado para o ensemble.

2.6.3.1 Medidas de similaridades entre componentes

Para estabelecer um índice de descorrelação entre dois candidatos a compor um ensemble, denotados por *i* e *j*, pode-se empregar uma medida de descorrelação amostral na forma:

$$ds_{ij} = 1 - \left| \frac{\sigma_{ij}}{\sigma_i \sigma_j} \right|, \tag{2.7}$$

com:

$$\sigma_i = \sqrt{\frac{1}{N-1} \sum_{k=1}^{N} \left(f_i(\mathbf{x}_k) - \bar{f}_i \right)^2}, \tag{2.8}$$

$$\sigma_j = \sqrt{\frac{1}{N-1} \sum_{k=1}^{N} \left(f_j(\mathbf{x}_k) - \bar{f}_j \right)^2}, \tag{2.9}$$

$$\sigma_{ij} = \frac{1}{N-1} \sum_{k=1}^{N} \left(f_i(\mathbf{x}_k) - \bar{f}_i \right)\left(f_j(\mathbf{x}_k) - \bar{f}_j \right), \tag{2.10}$$

onde N é a cardinalidade do conjunto de dados, $f_i(\mathbf{x}_k)$ é a saída do candidato i para a amostra k e $\bar{f}_i$ é a média das saídas produzidas pelo candidato i. A medida de descorrelação ds_{ij} excursiona no intervalo [0,1].

Uma outra possibilidade é o emprego do divergente de Kullback-Leibler (COVER & THOMAS, 1991; KULLBACK & LEIBLER, 1951), também chamado de *entropia relativa* ou *entropia cruzada*. Sejam p e q duas distribuições, então a distância entre elas, a partir de N amostras extraídas das respectivas distribuições, é definida como segue:

$$D(p,q) = \sum_{i=1}^{N} p_i \log\left(\frac{p_i}{q_i} \right). \tag{2.11}$$

Entretanto, $D(p,q)$ não é uma medida de distância no sentido amplo, devido ao fato de não ser simétrica, isto é, $D(p,q) \neq D(q,p)$. Para corrigir este problema, pode-se definir a entropia relativa simétrica como segue:

$$D(p,q) = \sum_{i=1}^{N} \left(p_i \log\left(\frac{p_i}{q_i} \right) + q_i \log\left(\frac{q_i}{p_i} \right) \right). \tag{2.12}$$

A similaridade é maior quanto menor for $D(p,q)$.

2.6.3.2 Métodos de seleção para classificação

Suponha que se deseja implementar um ensemble dispondo de M candidatos a comporem o ensemble, visando aproximar uma função f: $\Re^n \rightarrow U$, onde U é o conjunto de todas as possíveis classes $\{1,...,K\}$. A predição associada a cada componente do ensemble será combinada por meio de voto majoritário e cada componente vota para uma classe ou então para nenhuma delas, o que pode ser representado por uma classe (K+1). O rótulo da classe que receber o maior número de votos é considerado como saída do ensemble.

Considere que U contém somente dois rótulos de classe, isto é, a função a ser aproximada é f: $\Re^n \rightarrow U$, com $U = \{-1,+1\}$, embora a dedução a seguir possa também ser generalizada para outras situações que não classificação binária (ZHOU *et al.*, 2002). Agora, suponha que estão disponíveis N amostras rotuladas, de modo que a saída esperada para estas amostras é dada na forma: $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \cdots & y_N \end{bmatrix}^T$, com $\mathbf{y} \in U^N$, e a saída produzida pelo componente i assume a forma: $\mathbf{f}_i = \begin{bmatrix} f_{i1} & f_{i2} & \cdots & f_{iN} \end{bmatrix}^T$, onde f_{ij} denota a saída do i-ésimo componente para a j-ésima amostra e $\mathbf{f}_i \in U^N$, i=1,...,M. Caso a saída atual do i-ésimo componente para a j-ésima amostra seja correta, então $f_{ij} y_j = +1$. Caso contrário, $f_{ij} y_j = -1$. Assim, o erro de generalização do i-ésimo componente é dado por:

$$e_i = \frac{1}{N} \sum_{j=1}^{N} Erro\left(f_{ij} y_j\right), \tag{2.13}$$

onde $Erro(x)$ é uma função definida na forma:

$$Erro(x) = \begin{cases} 1 & se \quad x = -1 \\ 0 & se \quad x = 1 \end{cases}. \tag{2.14}$$

Defina um vetor soma como $\mathbf{soma} = \begin{bmatrix} soma_1 & soma_2 & \cdots & soma_N \end{bmatrix}^T$, onde $soma_j$ denota a soma da saída atual de todos os componentes para a j-ésima amostra, isto é:

$$soma_j = \sum_{i=1}^{M} f_{ij} \ . \tag{2.15}$$

Então a saída do ensemble de M componentes para a j-ésima amostra é dada por:

$$\hat{f}_j = sign(soma_j)\,, \tag{2.16}$$

onde *sign*(*x*) é uma função definida como:

$$sign(x) = \begin{cases} 1 & se \;\; x \geq 0 \\ -1 & se \;\; x < 0 \end{cases}. \tag{2.17}$$

Se a saída atual do ensemble para a *j*-ésima amostra está correta, então $\hat{f}_j y_j = +1$. Se está errada, então $\hat{f}_j y_j = -1$. Logo, o erro de generalização do ensemble pode ser expresso como segue:

$$\hat{e} = \frac{1}{N}\sum_{j=1}^{N} Erro(\hat{f}_j y_j)\,. \tag{2.18}$$

Aqui, pode-se adotar duas hipóteses independentes, úteis na seleção dos componentes.

<u>Hipótese de poda</u>

Suponha que o *k*-ésimo componente é excluído do ensemble. Então a saída do novo ensemble para a *j*-ésima amostra assume a forma:

$$\hat{f}_j^{'} = sign(soma_j - f_{kj})\,, \tag{2.19}$$

e o erro de generalização do novo ensemble é:

$$\hat{e}' = \frac{1}{N}\sum_{j=1}^{N} Erro(\hat{f}_j^{'} y_j)\,. \tag{2.20}$$

Das equações (2.18) e (2.20), pode-se deduzir a equação (2.21) dada por:

$$\sum_{j=1}^{N}\left\{Erro(sign(soma_j)y_j) - Erro(sign(soma_j + f_{kj})y_j)\right\} \geq 0 \tag{2.21}$$

A equação (2.21) é satisfeita quando $\hat{e}$ é maior ou igual a $\hat{e}'$, significando que a exclusão do *k*-ésimo componente conduz a melhores resultados quando comparado ao ensemble original.

Então, considerando que a exclusão do *k*-ésimo componente irá impactar a saída do ensemble sobre a *j*-ésima amostra, onde $\left|soma_j\right| > 1$, e considerando as propriedades da combinação da função *Erro*(*x*) e *sign*(*x*) quando $x \in \{-1,+1\}$ e $z \in \{-1,+1\}$, isto é:

$$Erro(sign(x)) - Erro(sign(x-z)) = -\frac{1}{2} sign(x+z), \tag{2.22}$$

obtém-se a condição para a extração do *k*-ésimo componente do ensemble:

$$\sum_{\substack{j=1 \\ j \in \{j:|soma_j| \leq 1\}}}^{N} sign((soma_j + f_{kj}) y_j) \leq 0 \tag{2.23}$$

Quando a equação (2.23) é satisfeita, indica que o número de componentes do ensemble pode ser reduzido promovendo um incremento na capacidade de generalização. Os componentes a serem excluídos do ensemble satisfazem a equação (2.23). Esta estratégia é conhecida como *método de poda* (ZHOU *et al.*, 2002).

Hipótese construtiva

Esta formulação constitui-se em uma adaptação do método apresentado por PERRONE & COOPER (1993) para tratar problemas de regressão. Suponha que o *k*-ésimo componente é incluído no ensemble, então a saída do novo ensemble para a *j*-ésima amostra assume a forma:

$$\hat{f}_j^{'} = sign(soma_j + f_{kj}), \tag{2.24}$$

e o erro de generalização do novo ensemble é dado por:

$$\hat{e}' = \frac{1}{N} \sum_{j=1}^{N} Erro(\hat{f}_j^{'} y_j). \tag{2.25}$$

Das equações (2.18) e (2.25) deduz-se que, se a equação (2.26) na forma:

$$\sum_{j=1}^{N} \left\{ Erro(sign(soma_j + f_{kj}) y_j) - Erro(sign(soma_j) y_j) \right\} \geq 0 \tag{2.26}$$

é satisfeita, então $\hat{e}$ é maior ou igual a $\hat{e}'$, significando que a inclusão do *k*-ésimo componente conduz a melhores resultados quando comparado ao ensemble original.

Então, considerando que a inclusão do *k*-ésimo componente irá impactar a saída do ensemble sobre a *j*-ésima amostra, onde $\left|soma_j\right| > 1$, e considerando as propriedades da combinação da função *Erro*(*x*) e *sign*(*x*) quando $x \in \{-1,+1\}$ e $z \in \{-1,+1\}$, isto é:

$$Erro(sign(x+z)) - Erro(sign(x)) = \frac{1}{2} sign(x-z), \tag{2.27}$$

obtém-se a condição para a inclusão do *k*-ésimo componente no ensemble:

$$\sum_{\substack{j=1 \\ j\in\{j:|soma_j|\leq 1\}}}^{N} sign((soma_j - f_{kj})y_j) \geq 0\,. \tag{2.28}$$

Quando a equação (2.28) é satisfeita, ela indica que o número de componentes do ensemble pode ser aumentado promovendo um incremento na capacidade de generalização. Os componentes a serem incluídos no ensemble devem satisfazer a equação (2.28). Esta estratégia é conhecida como *método construtivo*.

2.6.3.3 Métodos de seleção para regressão

Suponha que existam *M* candidatos disponíveis para compor um ensemble com o objetivo de aproximar uma função f: $\Re^n \rightarrow \Re^d$. Considere também que a combinação vai se dar por média simples ou média ponderada, em que os pesos α_i (i = 1,2,..., M) podem ser obtidos de acordo com as propostas de HASHEM (1997).

Tratando apenas do caso em que d = 1, ou seja, uma única saída, e sendo a entrada **x** amostrada de acordo com uma distribuição de probabilidade $p(\mathbf{x})$ e a saída esperada denotada por y(**x**), então a saída do ensemble para **x** é dada por:

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i f_i(\mathbf{x})\,. \tag{2.29}$$

O erro de generalização $e_i(\mathbf{x})$ do *i*-ésimo componente do ensemble sobre **x** é expresso como segue:

$$e_i(\mathbf{x}) = E\left[\left(f_i(\mathbf{x}) - y(\mathbf{x})\right)^2\right]\,, \tag{2.30}$$

e o erro de generalização $\hat{e}(\mathbf{x})$ do ensemble sobre **x** assume a forma:

$$\hat{e}(\mathbf{x}) = E\left[\left(\hat{f}(\mathbf{x}) - y(\mathbf{x})\right)^2\right]\,. \tag{2.31}$$

Logo, o erro de generalização do *i*-ésimo componente do ensemble, tomando a distribuição $p(\mathbf{x})$, admite a seguinte representação:

$$e_i = \int_X e_i(\mathbf{x})p(\mathbf{x})d\mathbf{x}\,, \tag{2.32}$$

onde X é o espaço de entrada.

De forma análoga, o erro de generalização do ensemble, tomando a distribuição $p(\mathbf{x})$, fica sendo:

$$\hat{e} = \int_X \hat{e}(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} . \tag{2.33}$$

O termo de correlação entre o *i*-ésimo e o *j*-ésimo componente é dado por:

$$C_{ij} = \int_X \left(f_i(\mathbf{x}) - y(\mathbf{x})\right)\left(f_j(\mathbf{x}) - y(\mathbf{x})\right) p(\mathbf{x}) d\mathbf{x} , \tag{2.34}$$

onde C_{ij} satisfaz as equações (2.35) e (2.36) abaixo:

$$C_{ii} = e_i , \tag{2.35}$$

$$C_{ij} = C_{ji} . \tag{2.36}$$

Considerando as equações (2.29) e (2.31), obtém-se:

$$\hat{e}(x) = E\left[\left(\sum_{i=1}^{M} \alpha_i f_i(\mathbf{x}) - y(\mathbf{x})\right)^2\right] . \tag{2.37}$$

Então, considerando as equações (2.33), (2.34) e (2.37) e a restrição $\sum_{i=1}^{M} \alpha_i = 1$, resulta:

$$\hat{e} = \sum_{i=1}^{M} \sum_{j=1}^{M} \alpha_i \alpha_j C_{ij} . \tag{2.38}$$

Supondo agora que todos os componentes têm pesos iguais, isto é, $\alpha_i = 1/M$ (i = 1,2,...,M), as predições dos componentes são combinadas via média simples. Então a equação (2.38) torna-se:

$$\hat{e} = \sum_{i=1}^{M} \sum_{j=1}^{M} C_{ij} / M^2 \tag{2.39}$$

A equação (2.39) mostra que o erro de generalização do ensemble depende da correlação que existe entre os componente. Portanto, métodos de seleção devem ser adotados de forma a garantir que a correlação entre os componentes seja bastante pequena. Aqui, podem ser adotadas duas hipóteses independentes, as quais serão úteis na seleção dos componentes.

Hipótese de poda

Esta formulação constitui-se em uma adaptação do método apresentado por ZHOU *et al.* (2002) para tratar problemas de classificação. Supondo que o *k*-ésimo componente seja excluído do ensemble. Então o erro de generalização do novo ensemble é:

$$\hat{e}' = \sum_{\substack{i=1 \\ i\neq k}}^{M} \sum_{\substack{j=1 \\ j\neq k}}^{M} C_{ij} \,/(M-1)^2 \,. \tag{2.40}$$

A partir das equações (2.39) e (2.40), obtém-se a equação (2.41):

$$\hat{e} \leq \frac{\left(2\sum_{\substack{i=1 \\ i\neq k}}^{M} C_{ik} + e_k \right)}{(2M-1)}, \tag{2.41}$$

que é satisfeita quando $\hat{e}$ é menor ou igual a $\hat{e}'$. Isto significa que o ensemble cujo *k*-ésimo componente foi excluído apresenta um desempenho superior. Logo, considerando a equação (2.41) junto com a equação (2.39), a restrição a ser satisfeita para justificar a exclusão de um componente *k* do ensemble é dada na forma:

$$(2M-1)\sum_{i=1}^{M}\sum_{j=1}^{M} C_{ij} \leq 2M^2 \sum_{\substack{i=1 \\ i\neq k}}^{M} C_{ik} + M^2 e_k \,. \tag{2.42}$$

A equação (2.42) indica que o número de componentes do ensemble pode ser reduzido visando um incremento na capacidade de generalização.

Logo, quando vários candidatos estão disponíveis, a utilização de parte do conjunto de candidatos pode produzir melhores resultados que a combinação de todos eles. Os componentes que deveriam ser excluídos do ensemble satisfazem a equação (2.42).

Hipótese construtiva

Supondo que o *k*-ésimo componente será incluído num ensemble que já possui *M* componentes, então o erro de generalização do novo ensemble assume a forma:

$$\hat{e}' = \sum_{i=1}^{M+1} \sum_{j=1}^{M+1} C_{ij} \,/(M+1)^2 \,. \tag{2.43}$$

Das equações (2.39) e (2.43), obtém-se a equação (2.44) a seguir:

$$\hat{e} \leq \frac{\left(2\sum_{\substack{i=1 \\ i \neq k}}^{M} C_{ik} + e_k \right)}{(2M+1)}, \tag{2.44}$$

que é satisfeita quando $\hat{e}$ é maior que $\hat{e}'$. Isto significa que o ensemble cujo *k*-ésimo componente foi incluído apresenta um desempenho superior. Logo, considerando a equação (2.44) junto com a equação (2.39), a restrição a ser satisfeita para justificar a inclusão de um componente *k* no ensemble é dada na forma:

$$(2M+1)\sum_{i=1}^{M}\sum_{j=1}^{M} C_{ij} \leq 2M^2 \sum_{\substack{i=1 \\ i \neq k}}^{M} C_{ik} + M^2 e_k . \tag{2.45}$$

A equação (2.45) indica que o número de componentes do ensemble pode ser aumentado visando um incremento na capacidade de generalização. Esta estratégia é conhecida como *método construtivo* (PERRONE & COOPER, 1993).

2.7 Considerações Finais

Neste capítulo, foi descrito o estado da arte da abordagem ensemble, cujo objetivo é incrementar o desempenho de classificadores e regressores. Vários métodos de geração, seleção e combinação foram apresentados dentro de uma estrutura unificada. Uma metodologia para construção de ensemble baseada em 3 passos (geração, seleção e combinação de componente) foi apresentada e vários aspectos relevantes para cada passo foram discutidos. No Capítulo 5, a abordagem ensemble será estendida para máquinas de vetores-suporte, na tentativa de agrupar as vantagens advindas de ambas as estratégias.

Capítulo 3

Mistura Hierárquica de Especialistas

Resumo: Há cerca de 10 anos, uma arquitetura modular para aprendizado supervisionado conhecida como mistura de especialistas (MEs, do inglês *Mixture of Experts*) foi devidamente formalizada (JACOBS *et al.*, 1991; JORDAN & JACOBS, 1994) e vem sendo aplicada com sucesso junto a uma ampla gama de problemas em aprendizado de máquina. Uma mistura de especialistas tenta solucionar problemas de classificação ou regressão com base em uma estratégia dividir-e-conquistar. Basicamente, o espaço de entrada é automaticamente dividido em regiões, sendo que para cada região existe um único ou um subconjunto de especialistas mais indicados a agir. A divisão de tarefas tende a ser suave e sobreposições de regiões de atuação são possíveis, a partir da implementação de uma rede *gating* que define os coeficientes da combinação convexa envolvendo a saída de cada especialista. Estes coeficientes devem ser sempre não-negativos e somar na unidade. Conceitos estatísticos são empregados na interpretação de MEs como modelos de mistura (MCLACHLAN & BASFORD, 1988), havendo uma densidade condicional associada a cada especialista, também denominado de componente da mistura. A abordagem dividir-e-conquistar vem sendo particularmente bem sucedida na atribuição dos especialistas para regimes diferentes em séries temporais estacionárias por partes (WEIGEND *et al.*, 1995), modelagem de descontinuidade em mapeamentos de entrada saída e em diversos tipos de problema de classificação (FRITSCH, 1997; MOERLAND, 1997; WATERHOUSE & ROBINSON,

1994). Neste capítulo, os principais conceitos de MEs serão apresentados na forma de uma rede neural modular e hierárquica para aprendizado supervisionado. Esta introdução se baseia na apresentação dada por JORDAN & JACOBS (1994), focalizando os problemas de classificação e regressão. O modelo estatístico será discutido em detalhes, a fim de motivar algumas possíveis extensões que serão propostas no Capítulo 6.

3.1 Introdução

Empregando a estratégia dividir-e-conquistar, JACOBS *et al.* (1991) e JORDAN & JACOBS (1994) propuseram uma arquitetura modular de redes neurais para aprendizado supervisionado conhecida como Mistura de Especialistas (MEs). Nesta proposta, cada especialista é um modelo linear, embora outros modelos de especialistas possam ser considerados, inclusive com a possibilidade de compor modelos lineares e não-lineares na mistura. Em linhas gerais, a participação de cada especialista é definida automaticamente, de acordo com a natureza do problema e com a habilidade intrínseca de cada um.

Embora possam existir aspectos do problema que serão tratados exclusivamente por um especialista, normalmente haverá uma sobreposição de atuação dos mesmos em regiões de transição, em que alguns especialistas vão ganhando mais importância, na medida em que outros vão perdendo.

A grande flexibilidade dos modelos de mistura está no fato de que os aspectos do problema a serem resolvidos por cada especialista, ou seja, a sua região de atuação, não são definidos a priori, sendo que a divisão de tarefas e a sintonia de cada especialista para melhorar o cumprimento de suas atribuições junto ao problema acabam sendo implementadas de forma interativa e com garantia de convergência, embora não necessariamente para um ótimo global.

Nesse tipo de abordagem, a função-objetivo é baseada na interpretação de MEs como modelos de mistura (MCLACHLAN & BASFORD, 1988), em que os coeficientes da mistura são obtidos a partir da definição de medidas de densidade de probabilidade condicional. Na seqüência, o modelo estatístico associado será discutido em detalhes, a fim

de motivar a apresentação de um método de aprendizado efetivo para a arquitetura, na forma de um algoritmo de Maximização da Esperança (EM, do inglês *Expectation Maximization*) (DEMPSTER *et al.*, 1977).

A idéia básica do algoritmo EM é que a maximização da verossimilhança pode ser simplificada se um conjunto de variáveis denominadas ausentes fossem conhecidas. O problema, entretanto, é que não se conhece a distribuição das variáveis ausentes. O ingrediente importante do algoritmo EM é, portanto, uma abordagem iterativa em dois passos: um passo *E*, no qual a esperança das variáveis ausentes é calculada (baseada nos valores atuais dos parâmetros), e um passo *M*, que maximiza esta função-objetivo completa (ou, equivalentemente, maximiza a verossimilhança). Em MOERLAND (1997) são descritos vários métodos para maximização da verossimilhança.

No final deste capítulo, são apresentadas algumas extensões adotadas em implementações práticas. No Capítulo 5, todos os resultados apresentados aqui serão utilizados para a incorporação de máquinas de vetores-suporte (SVMs – do inglês s*upport vector machines*) junto ao formalismo de mistura de especialistas.

3.2 Posicionamento da abordagem

Seguindo a abordagem proposta por JORDAN & JACOBS (1994), os conceitos de mistura de especialistas serão apresentados neste capítulo no contexto de redes neurais modulares, sujeitas a treinamento supervisionado, ou seja, sujeitas a aprendizado a partir de dados amostrados.

Arquiteturas de redes neurais, tais como perceptrons multicamadas, têm alcançado bons resultados como aproximadores universais (HORNIK *et al.*, 1989) e têm sido empregadas em muitos problemas distintos, desde classificação de padrões até controle de processos. Enquanto há ainda muitas questões técnicas e conceituais a serem elucidadas, particularmente junto à definição da dimensão mais apropriada da arquitetura e junto aos algoritmos de treinamento, há considerável incentivo também para se investir na proposição de novas arquiteturas ou arquiteturas especificamente projetadas para uma classe bem

definida de problemas. Idealmente, tais arquiteturas deveriam ser motivadas estatisticamente e ter parâmetros que são facilmente interpretáveis, além de permitir aumentar a eficiência dos algoritmos de treinamento, viabilizando aplicações em tempo real.

Motivados por tais preocupações, vários pesquisadores têm investigado aproximadores universais que incorporam conjuntamente resultados conceituais e práticos de estatística e de redes neurais. Uma tendência marcante em tais trabalhos é o uso de modelos separados para tratar aspectos distintos de um mesmo problema. A abordagem mais comum é definir algum critério para dividir *a priori* o problema a ser tratado em uma série de sub-problemas e, na seqüência, atribuir um aproximador universal ou especialista para solucionar cada um deles. As várias abordagens já propostas diferem na técnica para realizar a divisão do problema e no cálculo da melhor solução global a partir das respostas locais dos especialistas.

Por outro lado, em lugar de realizar uma divisão *a priori* do problema e só então buscar a solução isolada de cada módulo constituinte, a arquitetura de mistura de especialistas emprega métodos probabilísticos de forma a promover uma abordagem sistemática capaz de definir automaticamente tanto a divisão de tarefas como a forma de combinar as soluções modulares, conforme elas vão sendo produzidas para cada proposta de divisão de tarefas.

Uma mistura de especialistas (JORDAN & JACOBS, 1994) é, portanto, um modelo probabilístico que pode ser interpretado como um modelo de mistura para estimação da distribuição de probabilidade condicional dos dados. O modelo consiste de uma rede *gating*, responsável pela divisão do problema em sub-problemas, e de redes especialistas, as quais vão se especializando no tratamento de cada um destes sub-problemas. Em termos de modelo de mistura, as redes especialistas correspondem à densidade de componente condicional e a rede *gating*, aos coeficientes da mistura dependentes da entrada. Esta interpretação de mistura de especialistas como modelo de mistura possibilita treinar os especialistas com o algoritmo de maximização da esperança (JORDAN & JACOBS, 1994).

Os modelos de mistura de especialistas também se apresentam em versões hierárquicas e, neste aspecto, são similares aos modelos CART (BREIMAN *et al.*, 1984),

MARS (FRIEDMAN, 1991) e ID3 (QUILAN, 1986). Entretanto, estes três modelos mencionados solucionam o problema de classificação ou regressão pela divisão explícita do espaço de entrada em sub-regiões, tal que somente um único especialista está contribuindo junto a cada sub-região, ao passo que na MEs a rede *gating* realiza uma transição gradativa entre os especialistas, permitindo inclusive que vários deles sejam selecionados para atuar em um determinado aspecto do problema ou região do espaço de entrada. Com esta separação severa do espaço de entrada, CART, MARS e ID3 tendem a ter um aumento na variância (JORDAN & JACOBS, 1994), especialmente no caso de espaços de entrada de alta dimensão, onde os dados geralmente apresentam uma distribuição esparsa. Em contraste, a rede *gating* em uma mistura de especialistas é capaz de propor uma separação suave do espaço de entrada, permitindo que múltiplos especialistas contribuam na definição da saída para um determinado dado de entrada, sem descartar a possibilidade de alocar um único especialista para algumas situações específicas. Com a possível participação de vários especialistas, criam-se condições para um decrescimento da variância (JORDAN & JACOBS, 1994).

Ainda referente à versão hierárquica de mistura de especialistas, as propostas originais definiram que tanto as redes especialistas quanto a rede *gating* realizam mapeamento linear, podendo ter ou não uma não-linearidade apenas na saída, dependendo do problema. No caso de problemas de classificação com várias classes, a não-linearidade para o e a rede *gating* é escolhida geralmente como sendo a função *softmax* (JACOBS *et al.*, 1991), ao passo que no caso de regressão somente a *gating* tem uma não-linearidade na saída. A seleção da não-linearidade depende da interpretação probabilística do modelo e será descrita nas próximas seções.

Visto que a rede *gating* se ocupa da decomposição em pequenas tarefas, a escolha do modelo da *gating* é um aspecto importante de projeto. O modelo tradicional de mistura de especialistas tem um neurônio perceptron com função de ativação *softmax* como *gating* (JORDAN & JACOBS, 1994). Isto conduz a uma divisão do espaço de entrada por hiperplanos. Uma abordagem alternativa é empregar *gated experts* (WEIGEND *et al.*, 1995). Neste modelo, a rede *gating* é um perceptron multicamadas (MLP, do inglês *Multilayer Perceptron*) com função de ativação *softmax* na saída, possibilitando uma divisão mais

flexível do espaço de entrada. Uma terceira abordagem é dividir o espaço de entrada com hiper-elipsóides suaves usando funções gaussianas normalizadas (XU *et al.*, 1995), cada uma centrada na região de atuação de um especialista específico. Finalmente, quando a mistura de especialistas é hierárquica, compondo uma estrutura em árvore, as folhas ou nós terminais da árvore contêm as redes especialistas e os nós não-terminais da árvore contêm uma rede *gating*. Com isso, divisões complexas do espaço de entrada são possíveis mesmo com o emprego de redes *gating* elementares. Nas próximas seções, todas esta abordagens serão discutidas em detalhes.

3.3 Arquitetura de mistura de especialistas

A arquitetura modular a ser considerada neste estudo é apresentada na Figura 3.1, sendo composta por m módulos referidos como redes especialistas, cada um implementando um função parametrizada $\mu_i = f_i(\theta_i, \mathbf{x})$ da entrada **x** para a saída μ_i, onde θ_i é o vetor de parâmetros do especialista *i*. As saídas de cada uma das redes especialistas recebe uma interpretação probabilística considerando que o especialista gera a saída **y** com probabilidade $P(\mathbf{y} \mid \mathbf{x}, \theta_i)$, onde μ_i é a média da função densidade de probabilidade *P*.

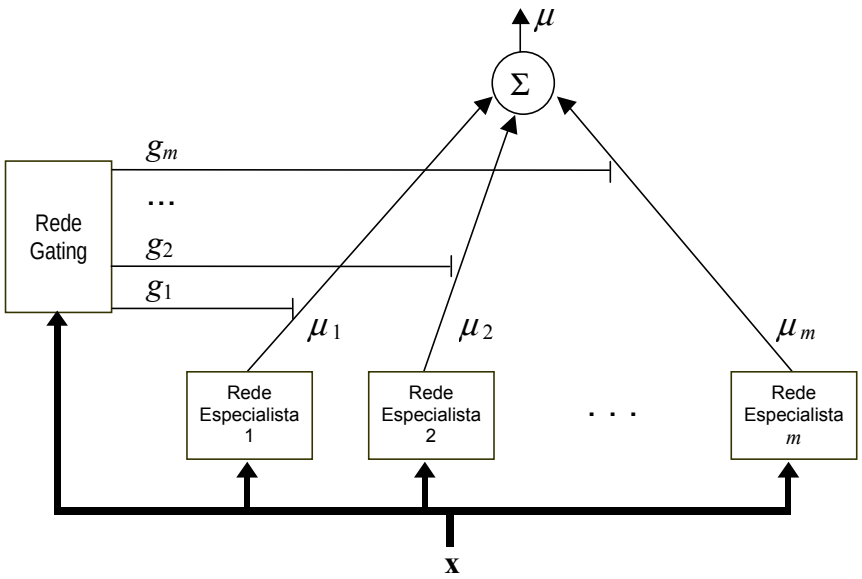


Figura 3.1 – Arquitetura de mistura de especialistas

Considerando que diferentes redes especialistas são apropriadas para diferentes regiões do espaço de entrada, a arquitetura requer um mecanismo capaz de identificar, para cada entrada **x**, o especialista ou combinação de especialistas mais capazes de produzir a saída correta, em termos probabilísticos. Isto é realizado por meio de uma rede auxiliar, conhecida como rede *gating*.

A interpretação probabilística da rede *gating* é de um sistema que calcula, para cada especialista, a probabilidade dele gerar a saída desejada, com base apenas no conhecimento da entrada **x**. Estas probabilidades são expressas pelos coeficientes g_i (i=1,...,m), de modo que estes devem ser não-negativos e devem produzir sempre o valor unitário quando somados, para cada **x**. Estes coeficientes não são constantes fixas, mas variam em função da entrada **x**. Caso os coeficientes g_i (i=1,...,m) sejam constantes e as redes especialistas atuem junto a todos os aspectos do problema, resulta um *ensemble*, conforme mencionado no Capítulo 2.

Há muitas formas de garantir que os coeficientes g_i (i=1,...,m) atendam as restrições acima. Uma abordagem é utilizar a função *softmax* (JACOBS *et al.*, 1991). Nas próximas seções, serão apresentadas outras formas utilizadas na literatura. A função *softmax* define um conjunto de variáveis intermediárias ξ_i (i=1,...,m) como funções da entrada **x** e de um vetor de parâmetros $\mathbf{v}_i$ (i=1,...,m) na forma:

$$\xi_i = \xi_i(\mathbf{x}, \mathbf{v}_i)\,. \tag{3.1}$$

Com isso, os coeficientes g_i (i=1,...,m) podem ser definidos em termos de ξ_i (i=1,...,m) como segue:

$$g_i = \frac{\exp(\xi_i)}{\sum_{k=1}^{m}\exp(\xi_k)}\,. \tag{3.2}$$

A partir desta definição, os coeficientes g_i (i=1,...,m) passam a respeitar as restrições impostas, isto é, são não-negativos e, somados, produzem sempre o valor unitário, para cada **x**. Uma interpretação probabilística para as variáveis intermediárias ξ_i (i=1,...,m) é que elas pertencem a uma família exponencial de distribuições de probabilidade (JORDAN & JACOBS, 1994).

A seguir, será especificado o modelo de probabilidade adotado para a arquitetura de mistura de especialistas. Considere que o conjunto de treinamento $\chi = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\}_{t=1}^{N}$ é gerado da seguinte forma: dada uma entrada **x**, um especialista *i* é escolhido com probabilidade $P(i \mid \mathbf{x}, \mathbf{v}^0)$ (onde o sobrescrito "0" será usado para distinguir os valores reais dos parâmetros do modelo de probabilidade adotado daqueles estimados pela rede *gating* ou pela rede especialista). Dada a escolha do especialista e dada a entrada, a saída desejada **y** é suposta ser gerada de acordo com a probabilidade $P(\mathbf{y} \mid \mathbf{x}, \theta_i^0)$. Cada um dos pares de entrada-saída é suposto ser gerado independentemente.

Observe que uma dada saída pode ser gerada de *m* formas diferentes, correspondendo às *m* formas diferentes de escolha do especialista. Assim, a probabilidade total de geração de **y** a partir de **x** é dada pela soma sobre *i*, na forma:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta^0) = \sum_{i=1}^{m} P(i \mid \mathbf{x}, \mathbf{v}^0) P(\mathbf{y} \mid \mathbf{x}, \theta_i^0), \tag{3.3}$$

onde Θ^0 denota o vetor contendo todos os parâmetros, na forma $\Theta^0 = \left[\theta_1^0, \theta_2^0, \ldots, \theta_m^0, \mathbf{v}^0\right]^T$. A densidade na equação (3.3) é conhecida como *mistura de densidade* ou *função de verossimilhança*. É uma mistura de densidade no espaço de saída, condicionada à escolha da entrada, onde $P(i \mid \mathbf{x}, \mathbf{v}^0)$ é a probabilidade de se escolher o especialista *i*, dados a entrada **x** e o vetor de parâmetros $\mathbf{v}^0$,e $P(\mathbf{y} \mid \mathbf{x}, \theta_i^0)$ é a probabilidade deste especialista gerar a saída **y**, dado a entrada **x** e o vetor de parâmetros θ_i^0.

É tarefa da rede *gating* modelar a probabilidade $P(i \mid \mathbf{x}, \mathbf{v}^0)$. É possível parametrizar esta probabilidade via equações (3.1) e (3.2), fazendo a saída da rede *gating* g_i (*i*=1,...,*m*) ser igual a $P(i \mid \mathbf{x}, \mathbf{v}_i)$.

A saída da mistura de densidades pode ser calculada através da média condicional. A média condicional $\mu = \mathrm{E}\left[P(\mathbf{y} \mid \mathbf{x}, \Theta^0)\right]$ é obtida tomando o valor esperado da equação (3.3):

$$\mu = \sum_{i=1}^{m} g_i \mu_i, \tag{3.4}$$

onde μ_i é a média condicional associada com a distribuição de probabilidade $P(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta}_i^0)$. Esta combinação convexa da saída do especialista é uma escolha usual para a saída de uma arquitetura modular.

3.4 Algoritmo de aprendizado baseado no gradiente para mistura de especialistas

Para desenvolver um algoritmo de estimação dos parâmetros de uma arquitetura de mistura de especialistas, é empregado o princípio de maximização da verossimilhança, definida na equação (3.3). Como é comumente adotado em estatística, é mais conveniente trabalhar com o logaritmo da verossimilhança que com a própria verossimilhança. Tomando o logaritmo de *m* densidades na forma da equação (3.3), e dado o conjunto de treinamento $\boldsymbol{\chi} = \{(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})\}_{t=1}^{N}$ chega-se à seguinte medida de verossimilhança:

$$l(\boldsymbol{\chi}, \boldsymbol{\Theta}) = \sum_{t=1}^{N} \log \sum_{i=1}^{m} P(i \mid \mathbf{x}^{(t)}, \mathbf{v}) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i) . \tag{3.5}$$

Uma abordagem para maximizar o logaritmo da verossimilhança é usar o gradiente ascendente (a abordagem mais indicada, no entanto, é usar o algoritmo EM, conforme será apresentado na seção 3.8). Calculando o gradiente de $l(\cdot,\cdot)$ com respeito a μ_i e ξ_i, resultam:

$$\frac{\partial l}{\partial \mu_i} = \sum_{t=1}^{N} h_i^{(t)} \frac{\partial}{\partial \mu_i} \log P(\mathbf{y}^{(t)} \mid, \mathbf{x}^{(t)}, \theta_i) \tag{3.6}$$

e

$$\frac{\partial l}{\partial \xi_i} = \sum_{t=1}^{N} \left(h_i^{(t)} - g_i^{(t)} \right), \tag{3.7}$$

onde $h_i^{(t)}$ é definida como $P(i \mid \mathbf{x}^{(t)}, \mathbf{y}^{(t)})$. Na derivação deste resultado, foi empregada a regra de Bayes:

$$P(i \mid \mathbf{x}^{(t)}, \mathbf{y}^{(t)}) = \frac{P(i \mid \mathbf{x}^{(t)}) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i)}{\sum_{j=1}^{m} P(j \mid \mathbf{x}^{(t)}) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_j)} . \tag{3.8}$$

Isto sugere que $h_i^{(t)}$ é definida como a probabilidade *a posteriori* do *i*-ésimo especialista, condicionada à entrada $\mathbf{x}^{(t)}$ e à saída $\mathbf{y}^{(t)}$. De modo equivalente, a probabilidade $g_i^{(t)}$ pode ser interpretada como a probabilidade *a priori* $P(i,\mathbf{x}^{(t)})$, ou seja, a probabilidade da *gating* escolher o *i*-ésimo especialista, dada somente a entrada $\mathbf{x}^{(t)}$. Com estas definições, a equação (3.8) pode ser interpretada com uma forma de aproximar a probabilidade *a posteriori* utilizando a probabilidade *a priori*.

Um caso especial interessante é uma arquitetura na qual as redes especialistas e a rede *gating* são modelos lineares e a densidade de probabilidade associada com os especialistas é uma gaussiana com matriz de covariância igual à identidade (válido somente para problemas de regressão). Neste caso, são obtidas as seguintes equações para aprendizado *on-line* (*on-line* implica apenas na ausência do somatório sobre *t*) (JORDAN & JACOBS, 1994):

$$\theta_i^{(k+1)} = \theta_i^{(k)} + \rho h_i^{(t)} (\mathbf{y}^{(t)} - \mu_i^{(t)}) \mathbf{x}^{(t)^T} \tag{3. 9}$$

$$\mathbf{v}_i^{(k+1)} = \mathbf{v}_i^{(k)} + \rho (h_i^{(t)} - g_i^{(t)}) \mathbf{x}^{(t)^T} \tag{3.10}$$

onde ρ é a taxa de aprendizado. Observe que ambas as equações acima têm a forma da regra dos quadrados mínimos (LMS, do inglês *Least Mean Squares*) clássica (WIDROW & STEARNS, 1985), com a atualização dos especialistas na equação (3.9) sendo modulada pela sua probabilidade *a posteriori* ($h_i^{(t)}$).

É também de interesse examinar a expressão para a probabilidade *a posteriori* no caso em que $P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i)$ é gaussiana (JORDAN & JACOBS, 1994):

$$h_i^{(t)} = \frac{g_i^{(t)} \exp\left\{-\frac{1}{2}\left(\mathbf{y}^{(t)} - \mu_i^{(t)}\right)^T \left(\mathbf{y}^{(t)} - \mu_i^{(t)}\right)\right\}}{\sum_{j=1}^{m} g_j^{(t)} \exp\left\{-\frac{1}{2}\left(\mathbf{y}^{(t)} - \mu_j^{(t)}\right)^T \left(\mathbf{y}^{(t)} - \mu_j^{(t)}\right)\right\}}. \tag{3.11}$$

Esta é uma medida da distância normalizada que reflete a magnitude relativa dos resíduos $\left(\mathbf{y}^{(t)} - \mu_i^{(t)}\right)$. Se o resíduo para o especialista *i* é pequeno em relação aos resíduos dos demais especialistas, então $h_i^{(t)}$ é grande. Caso contrário, $h_i^{(t)}$ é pequeno. Observe que, além disto, os $h_i^{(t)}$ (i=1,...,m) são não-negativos e somados produzirem sempre o valor unitário,

para cada $\boldsymbol{x}^{(t)}$. Isto implica que o crédito é distribuído para os especialistas de uma maneira competitiva.

É possível utilizar outros membros da família exponencial de densidade como densidade de componente para os especialistas, para permitir que os parâmetros de dispersão (isto é, covariância) sejam incorporados ao modelo, os quais são estimados via algoritmo de aprendizado. Nas próximas seções, será apresentado como incorporar tais parâmetros.

3.5 Arquitetura de mistura hierárquica de especialistas (HMEs)

A arquitetura de mistura de especialistas apresentada na Figura 3.1 soluciona problemas complexos de classificação e regressão, alocando os diferentes especialistas para tratar diferentes regiões do espaço de entrada. Esta abordagem pode apresentar vantagens para problemas em que os especialistas são bastante simples, quando comparados a um único especialista responsável por solucionar o problema como um todo. No entanto, quando pelo menos um dos especialistas que compõem a mistura tiver alocado a si uma tarefa que não é tão simples de resolver, é possível repetir o mesmo argumento e substituir este especialista por uma mistura de especialistas, resultando em uma mistura de mistura de especialistas. E o mesmo argumento pode ser aplicado de modo recorrente, enquanto houver especialistas suficientemente complexos a ponto de admitir uma divisão de suas tarefas em sub-tarefas menores e mais simples. Isto sugere uma abordagem dividir-e-conquistar aninhada, em que uma arquitetura estruturada em árvore é usada para realizar separações hierárquicas do espaço de entrada (ver Figura 3.2). O processo de divisão hierárquica termina em uma árvore de especialistas, os quais, por serem definidos sobre regiões relativamente pequenas do espaço de entrada, podem ajustar funções simples para os dados (por exemplo, funções lineares). Esta arquitetura hierárquica, sugerida por JORDAN & JACOBS (1994), tem um vínculo muito próximo com modelos em árvore para classificação e regressão em aprendizado de máquina (BREIMAN *et al.*, 1984). De fato, a

arquitetura de JORDAN & JACOBS (1994) pode ser vista como uma versão probabilística de tais modelos.

A estrutura matemática fundamental da arquitetura para mistura hierárquica de especialistas (HMEs, do inglês *Hierarchical Mixture of Experts*) é essencialmente a mesma daquela para mistura de especialistas. O modelo de probabilidade é diretamente estendido para permitir uma seqüência aninhada de especialistas a serem escolhidos, correspondendo aos caminhos entre a raiz e as folhas de uma árvore.

Considere uma HMEs de duas camadas na forma de uma arvore binária, conforme mostrado na Figura 3.2. Cada rede especialista (*i,j*) produz sua saída a partir da entrada **x** de acordo com:

$$\mu_{ij} = f(\theta_{ij}\mathbf{x}), \tag{3.12}$$

onde θ_{ij} é o vetor de parâmetros e f pode ser uma função linear ou não linear. O vetor de entrada **x** considerado possui uma entrada adicional e constante, de valor unitário, denominada entrada de polarização.

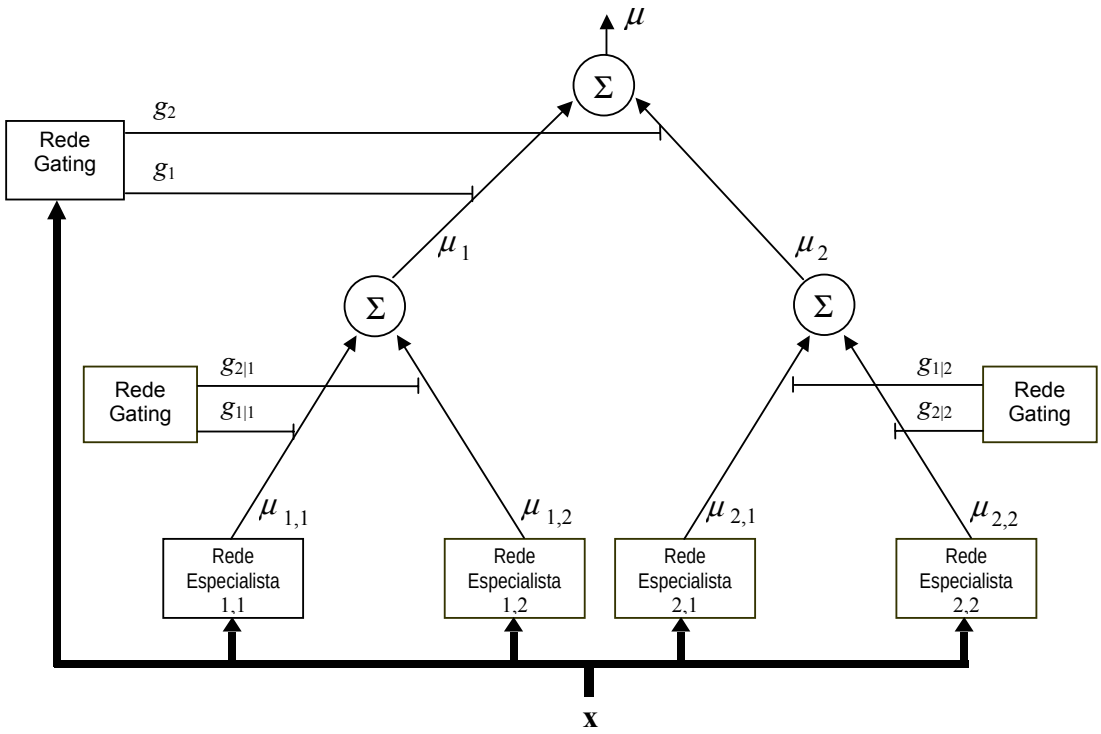


Figura 3.2: Arquitetura de uma mistura hierárquica de especialistas

No caso de problemas de classificação, uma vez que as redes *gating* executam classificação multimodos[1] entre os especialistas, a não-linearidade pode ser escolhida como *softmax* (JACOBS *et al.*, 1991). Os valores de saída da rede *gating* no nível superior são calculados de acordo com a expressão a seguir:

$$g_i = \frac{\exp(\xi_i)}{\sum_{k=1}^{m} \exp(\xi_k)} \text{ com } \xi_i = \xi_i(\mathbf{x}, \mathbf{v}_i). \tag{3.13}$$

Devido à forma especial da não-linearidade *softmax*, os coeficientes g_i (i=1,...,m) são não-negativos e somados produzirem sempre o valor unitário, para cada **x**. Estes coeficientes podem ser interpretados como a probabilidade condicional local de que um vetor de entrada **x** reside na região dos nós-filhos associados. A rede *gating* do nível inferior calcula a ativação de sua saída similar à rede *gating* do nível superior, na forma:

$$g_{j|i} = \frac{\exp(\xi_{ij})}{\sum_{k=1}^{m} \exp(\xi_{ik})} \text{ com } \xi_{ij} = \xi_{ij}(\mathbf{x}, \mathbf{v}_{ij}). \tag{3.14}$$

As ativações da saída das redes especialistas são ponderadas pela ativação da saída da rede *gating*, quando elas se propagam até a parte superior da árvore para formar o vetor de saída global. Especificamente, a saída do i-ésimo nó interno na segunda camada da árvore[2] é:

$$\mu_i = \sum_{j=1}^{m} g_{j|i} \mu_{ij}, \tag{3.15}$$

e a saída do nível superior (nó raiz) é

$$\mu = \sum_{j=1}^{r} g_i \mu_j. \tag{3.16}$$

Visto que tanto os especialistas como a rede *gating* calculam sua ativação como uma função da entrada **x**, a saída global da arquitetura é uma função não-linear da entrada (mesmo no caso de especialistas lineares). Além disso, diferentes espaços de entrada podem ser usados pela rede *gating* e pelo especialista. No caso de reconhecimento de fala, por exemplo, a rede *gating* pode ser suprida com características de entradas adicionais, a fim de facilitar a discriminação entre diferentes sons (FRITSH, 1996).

[1] É o mesmo que classificação com múltiplas classes

[2] Considerando uma árvore com fator de ramificação igual a r, com m folhas cada nó e o número de níveis igual a l. Neste caso particular l = 2.

O modelo de probabilidade para uma árvore de dois níveis (conforme Figura 3.2) é o seguinte:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_{i=1}^{r} P(i \mid \mathbf{x}, \mathbf{v}_i) \sum_{j=1}^{m} P(j \mid i, \mathbf{x}, \mathbf{v}_{ij}) P(\mathbf{y} \mid \mathbf{x}, \theta_{ij}), \tag{3.17}$$

que corresponde a uma escolha do rótulo *i* com probabilidade $P(i \mid \mathbf{x}, \mathbf{v}_i)$, seguida por uma escolha do rótulo *j* com probabilidade $P(j \mid i, \mathbf{x}, \mathbf{v}_{ij})$. Este modelo de probabilidade produz o seguinte logaritmo da função de verossimilhança (JORDAN & JACOBS, 1994):

$$l(\chi, \Theta) = \sum_{t=1}^{N} \log \sum_{i=1}^{r} P(i \mid \mathbf{x}^{(t)}, \mathbf{v}_i) \sum_{j=1}^{m} P(j \mid i, \mathbf{x}^{(t)}, \mathbf{v}_{ij}) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij}), \tag{3.18}$$

onde, como no caso das probabilidades *a priori* $g_i^{(t)} = P(i \mid \mathbf{x}^{(t)}, \mathbf{v}_i)$ e $g_{j|i}^{(t)} = P(j \mid i, \mathbf{x}^{(t)}, \mathbf{v}_{ij})$, o cálculo se dá em termos das variáveis fundamentais ξ_i e ξ_{ij}, usando a função *softmax* (equações 3.13 e 3.14). A regra de Bayes é também empregada aqui para definir a probabilidade *a posteriori* na forma:

$$h_i^{(t)} = \frac{g_i^{(t)} \sum_{j=1}^{m} g_{j|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{k=1}^{r} g_k^{(t)} \sum_{j=1}^{m} g_{j|k}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{kj})}, \tag{3.19}$$

$$h_{j|i}^{(t)} = \frac{g_{j|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{k=1}^{m} g_{k|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ik})}. \tag{3.20}$$

A probabilidade *a posteriori* $h_i^{(t)}$ pode ser vista como o crédito atribuído ao *i*-ésimo nó não-terminal da rede, e a probabilidade *a posteriori* $h_{j|i}^{(t)}$ é o crédito atribuído aos ramos acima dos nós não-terminais. O produto é, portanto, o crédito atribuído ao especialista (*i*,*j*). Um relacionamento recursivo é útil para calcular eficientemente a probabilidade *a posteriori* quando muitos níveis hierárquicos estão envolvidos. Esta recursão foi implementada durante o desenvolvimento desta tese.

3.6 Aprendizado por gradiente ascendente para HMEs

Após definir a verossimilhança do modelo para um dado conjunto de treinamento, é possível tratar o aprendizado como um problema de maximização da verossimilhança, conforme introduzido por JORDAN & JACOBS (1994). A derivação deste algoritmo de aprendizado é razoavelmente simples e direta, podendo ser realizada tanto via método de aprendizado *on-line* quanto em lote.

3.6.1 Verossimilhança

A fim de derivar um algoritmo de atualização para os parâmetros da rede *gating* e de cada rede especialista, é necessário o uso das derivadas do logaritmo da verossimilhança (equação 3.18) com respeito aos parâmetros da rede *gating* e de cada especialista, respectivamente. No que segue, as derivações referem-se à Figura 3.2. Para a rede *gating* do nível superior, considerando modelos lineares, obtém-se:

$$\frac{\partial l(\chi,\Theta)}{\partial \mathbf{v}_k} = \sum_{t=1}^{N} \frac{\sum_{i=1}^{r} (\partial g_i / \partial v_k) \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})} \tag{3.21}$$

$$\frac{\partial l(\chi,\Theta)}{\partial \mathbf{v}_k} = \sum_{t=1}^{N} \frac{\sum_{i=1}^{r} g_i (\delta_{ik} - g_k) \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})} \mathbf{x}^{(t)} \tag{3.22}$$

$$\frac{\partial l(\chi,\Theta)}{\partial \mathbf{v}_k} = \sum_{t=1}^{N} \frac{g_k \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij}) - g_k \sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})} \mathbf{x}^{(t)} \tag{3.23}$$

$$\frac{\partial l(\chi,\Theta)}{\partial \mathbf{v}_k} = \sum_{t=1}^{N} (h_k - g_k) \mathbf{x}^{(t)} , \tag{3.24}$$

onde a derivada da função *softmax* é dada por:

$$\frac{\partial g_i}{\partial \xi_k} = g_i (\delta_{ik} - g_k) . \tag{3.25}$$

onde δ é função delta[3].

[3] $\delta_{ij} = \begin{cases} 0, & se \ \ i \neq j \\ 1, & se \ \ i = j \end{cases}$

Similarmente, pode-se demonstrar (JORDAN & JACOBS, 1994) que a derivada da verossimilhança com respeito à rede *gating* (considerando também modelos lineares) do nível superior é dada por:

$$\frac{\partial l(\chi,\Theta)}{\partial \mathbf{v}_{kl}} = \sum_{t=1}^{N} h_k (h_{j|k} - g_{j|k}) \mathbf{x}^{(t)} \ . \tag{3.26}$$

Devido ao interesse no conjunto de parâmetros que maximiza o logaritmo da verossimilhança dos dados observados, é então executado o gradiente ascendente no espaço dos parâmetros utilizando o gradiente da verossimilhança e um fator de aprendizado ρ para atualizar os parâmetros da rede *gating*, como segue:

$$\mathbf{v}_i^{(k+1)} = \mathbf{v}_i^{(k)} + \rho \sum_{t=1}^{N} (h_i - g_i) \mathbf{x}^{(t)} \ , \tag{3.27}$$

$$\mathbf{v}_{ij}^{(k+1)} = \mathbf{v}_{ij}^{(k)} + \rho \sum_{t=1}^{N} h_i (h_{j|i} - g_{j|i}) \mathbf{x}^{(t)} \ . \tag{3.28}$$

Cada uma destas derivadas parciais acima tem uma interpretação em termos de atribuição de crédito. A atualização dos parâmetros da rede *gating* tem como objetivo tornar o valor da probabilidade *a priori* tão próxima quanto possível do valor da probabilidade *a posteriori* (equações 3.27 e 3.28).

A regra do algoritmo acima sugere que uma atualização seja executada após a apresentação do conjunto de treinamento completo. Ao invés de calcular o gradiente da verossimilhança sobre todo o conjunto de treinamento, é possível também usar uma variação deste, chamada atualização por *gradiente estocástico*, que atualiza os parâmetros cada vez que um número fixo *k* de exemplos de treinamento tiver sido apresentado. Esta forma de atualização dos parâmetros é usualmente chamada de aprendizado *on-line* e conduz a uma rápida convergência (FRITSCH, 1996).

Resta derivar uma regra de atualização para a rede especialista. Dependendo do modelo de densidade de probabilidade escolhido para o especialista, são obtidas regras de atualização diferentes. Portanto, é necessário distinguir entre a tarefa de classificação e regressão, de modo a derivar algoritmos de atualização apropriados.

3.6.2 Atualização dos parâmetros do especialista para regressão

Quando a HMEs é utilizada para aproximação de funções, a densidade de probabilidade fundamental é tomada como sendo gaussiana. Para simplificar a derivação da regra de atualização, é adotada uma densidade gaussiana com variância unitária, embora a regra de atualização para gaussiana com matriz de covariância completa também exista (JORDAN & XU, 1995). O gradiente do logaritmo da verossimilhança com respeito ao (*k,l*)-ésimo especialista é dado por:

$$\frac{\partial l(\chi,\theta)}{\partial \theta_{kl}} = \sum_{t=1}^{N} \frac{g_k g_{l|k} \partial P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{kl}) / \partial \theta_{kl}}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{j|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}, \tag{3.29}$$

$$\frac{\partial l(\chi,\theta)}{\partial \theta_{kl}} = \sum_{t=1}^{N} h_{kl}(\mathbf{y}^{(t)} - \mu_{ij}^{(t)})\mathbf{x}^{(t)T} \ . \tag{3.30}$$

Sendo assim, pode-se empregar a regra de atualização para os parâmetros dos especialistas, atualizando toda a matriz de parâmetros (considerando modelos lineares) na forma:

$$\theta_{kl}^{(k+1)} = \theta_{kl}^{(k)} + \eta \sum_{t=1}^{N} h_{kl}(\mathbf{y}^{(t)} - \mu_{kl}^{(t)})\mathbf{x}^{(t)T} \tag{3.31}$$

3.6.3 Atualização dos parâmetros do especialista para classificação

No caso de classificação de padrões, o mesmo tipo de densidade de probabilidade é utilizada tanto para os especialistas quanto para a rede *gating*, uma vez que ambos realizam uma classificação multimodos. Para treinamento de classificadores, é usualmente considerado um conjunto de dados com saída desejada pertencente ao conjunto $\{0,1\}$, ou seja, há um rótulo de uma classe associado a cada vetor de entrada **x**. Sendo $\mathbf{y}_l^{(t)} = 1$ quando a classe *l* é a correta para o padrão $\mathbf{x}^{(t)}$, e $\mathbf{y}_l^{(t)} = 0$ caso contrário. Com o número total de classes sendo dado por *K*, a densidade de probabilidade *multinomial* pode ser expressa como segue:

$$P(y \mid x, \theta) = \prod_{m=1}^{K} p_m^{y_m} \ . \tag{3.32}$$

onde p_i é a probabilidade *multinomial* associada às diferentes classes (no caso dos nós-filhos), y_i são as saídas desejadas para cada classe (que são 0 para todas as saídas e 1 para a classe correta). Com densidade de probabilidade *multinomial*, uma escolha conveniente para função de ativação do especialista *j* é a função *softmax* (JACOBS *et al.*, 1991).

Usando este modelo de probabilidade simplificado, obtém-se a derivada do logaritmo de verossimilhança com respeito ao vetor de parâmetros da saída *q* na rede especialista (*k*,*l*), como segue:

$$\frac{\partial l(\chi,\theta)}{\partial \theta_{klq}} = \sum_{t=1}^{N} \frac{g_k g_{l|k} (\partial P(y^{(t)} \mid \mathbf{x}^{(t)}, \theta_{kl}) / \partial \theta_{klq})}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{l|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})} \tag{3.33}$$

$$\frac{\partial l(\chi,\theta)}{\partial \theta_{klq}} = \sum_{t=1}^{N} \frac{g_k g_{l|k} (\partial \mu_{klc}^{(t)} / \partial \theta_{klq})}{\sum_{i=1}^{r} g_i \sum_{j=1}^{m} g_{l|i} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})} \tag{3.34}$$

$$\frac{\partial l(\chi,\theta)}{\partial \theta_{klq}} = \sum_{t=1}^{N} h_{kl}^{(t)} (\delta_{lq} - \mu_{klq}) \mathbf{x}^{(t)} \tag{3.35}$$

$$\frac{\partial l(\chi,\theta_{kl})}{\partial \theta_{klm}} = \sum_{t=1, l=q}^{N} h_{kl}^{(t)} (1 - \mu_{klq}) \mathbf{x}^{(t)} - \sum_{t=1, l\neq q}^{N} h_{kl}^{(t)} \mu_{klq} \mathbf{x}^{(t)}, \tag{3.36}$$

que conduz à regra de atualização dos parâmetros do especialista dada por:

$$\theta_i^{(k+1)} = \theta_i^{(k+1)} + \rho(\sum_{t=1, l=q}^{N} h_{kl}^{(t)} (1 - \mu_{klq}) \mathbf{x}^{(t)} - \sum_{t=1, l\neq q}^{N} h_{kl}^{(t)} \mu_{klq} \mathbf{x}^{(t)}). \tag{3.37}$$

Novamente, as fórmulas de atualização podem ser aplicadas em lote ou *on-line*. A avaliação da regra de aprendizado do gradiente ascendente será postergada para depois das duas próximas seções, onde será derivado um algoritmo de aprendizado mais eficiente para a arquitetura HMEs.

3.7 Interpretação probabilística

As arquiteturas MEs e HMEs são melhor entendidas como árvores de decisão probabilística. Para cada vetor de entrada **x**, os valores de saída calculados pela rede *gating* são interpretados como a probabilidade *multinomial* de selecionar um dos nós-filhos.

Partindo do nó raiz, uma seqüência particular de decisões é realizada baseada na distribuição de probabilidade imposta pela rede *gating*. Este processo converge para um nó terminal da árvore contendo uma rede especialista específica. Esta rede especialista calcula, por exemplo, uma ativação linear μ_{ij} usando sua matriz de parâmetros. O vetor μ_{ij} é considerado a média da densidade de probabilidade que modelou a geração dos vetores de saída.

A parametrização da rede *gating* corresponde ao modelo de probabilidade *multinomial*, que é um caso especial de modelo linear generalizado (GLIM) (MCCULLAGH & NELDER, 1983). Ou seja, a saída da rede *gating* segue uma densidade *multinomial* conforme apresentado na equação (3.32).

A densidade de probabilidade para a rede especialista *i* é tomada como membro de uma família exponencial de densidade. No caso de regressão, o componente probabilístico é geralmente tomado como sendo a densidade gaussiana, produzindo:

$$P(\mathbf{y} \mid \mathbf{x}, \theta_i) = \frac{1}{(2\pi)^{1/2} \prod_i \sigma_i} \exp\left\{ -\frac{1}{2} \sum_i \frac{(\mathbf{x}_i - \mu_i)^2}{\sigma_i^2} \right\}. \tag{3.38}$$

Observa-se que a densidade gaussiana está sendo considerada como matriz de covariância diagonal, isto é, $\sigma\mathbf{I}$. Já no caso de classificação multimodos, a função densidade de probabilidade dos especialistas pode ser a mesma para a rede *gating*, com a diferença de que a rede *gating* discrimina entre os nós-filhos e a rede especialista discrimina entre classes de saída. Nas próximas seções, a densidade condicional de Bernoulli também será usada como densidade de probabilidade dos especialistas.

Com base nestas suposições, a probabilidade total de geração da saída **y** a partir da entrada **x** pode ser dada na forma de um modelo de mistura hierárquica, como segue:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_i g_i(\mathbf{x}, \mathbf{v}_i) \sum_j g_{j|i}(\mathbf{x}, \mathbf{v}_{ij}) P(\mathbf{y} \mid \mathbf{x}, \theta_{ij}). \tag{3.39}$$

Na notação, Θ contém os parâmetros das redes *gating* $\mathbf{v}_i$, $\mathbf{v}_{ij}$ e os parâmetros das redes especialistas θ_{ij}.

3.7.1 Probabilidade *a posteriori*

Nesta seção, será apresentado um equacionamento referente à probabilidade dos ramos *a posteriori* e dos nós. Considere o treinamento de uma dada arquitetura HMEs, onde se conhece explicitamente o vetor de saída desejada **y** para cada entrada **x**. Neste contexto, as probabilidades das redes *gating* g_i e $g_{j|i}$ são tomadas como probabilidades *a priori* dos ramos, pois elas são calculadas com base no vetor de entrada **x** somente, sem qualquer conhecimento do vetor de saída desejada **y**. Usando tanto o vetor de entrada quanto o vetor de saída, a probabilidade *a posteriori* dos ramos pode ser definida para as redes *gating* como segue:

$$h_i^{(t)} = \frac{g_i^{(t)} \sum_{j=1}^{m} g_{j|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{j=1}^{r} g_j^{(t)} \sum_{k=1}^{m} g_{j|k}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{jk})}, \tag{3.40}$$

$$h_{j|i}^{(t)} = \frac{g_{j|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{k=1}^{m} g_{k|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ik})}. \tag{3.41}$$

Baseado nestas probabilidades *a posteriori* condicionais, é possível calcular a probabilidade do nó não condicional, para cada nó da árvore, multiplicando todas as probabilidades *a posterori* condicionais dos ramos ao longo do caminho do nó raiz ao nó em questão. Desta forma, é possível atribuir uma probabilidade *a posteriori* para cada uma das redes especialistas, produzindo:

$$h_{ij}^{(t)} = h_i^{(t)} h_{j|i}^{(t)} = \frac{g_i^{(t)} g_{j|i}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})}{\sum_{l=1}^{r} g_l^{(t)} \sum_{k=1}^{m} g_{k|l}^{(t)} P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{lk})}, \tag{3.42}$$

onde h_{ij} é interpretada como a probabilidade da rede especialista (i,j) gerar o par de dados observáveis (**x**, **y**). Observe que, para o conjunto de teste, a probabilidade *a posteriori* não pode ser calculada, pois não há conhecimento a respeito do vetor de saída desejada **y**. Elas são empregadas apenas para a derivação do algoritmo de aprendizado.

3.8 Aprendizado EM

O algoritmo de aprendizado por maximização da esperança, proposto por DEMPSTER *et al.* (1977), é uma técnica geral para estimação do máximo da verossimilhança aplicado principalmente a aprendizado não-supervisionado, isto é, clusterização e estimação da mistura de densidades. No entanto, não há razões para que a estrutura EM não possa ser apropriadamente empregada junto a problemas de aprendizado supervisionado.

3.8.1 Algoritmo EM geral

JORDAN & JACOBS (1994) derivaram um algoritmo de maximização da esperança (EM, *expectation maximization* na literatura em inglês) para estimação dos parâmetros das arquiteturas MEs e HMEs. Este algoritmo é uma alternativa ao método do gradiente apresentado anteriormente e particularmente útil para modelos nos quais as redes especialistas e a rede *gating* têm uma forma paramétrica simples. Cada iteração do algoritmo EM consiste de duas fases: (1) uma propagação recursiva para cima e para baixo na árvore para calcular a probabilidade *a posteriori* (o passo *E*); e (2) uma solução de um conjunto de problemas de maximização da verossimilhança, ponderados nos nós não-terminais e terminais da árvore (o passo *M*). JORDAN & JACOBS (1994) testaram este algoritmo para problemas de identificação não-linear da dinâmica de um braço de robô com quatro graus de liberdade, e relataram uma rápida convergência. Em comparação com o algoritmo de retropropagação em uma rede perceptron multicamadas, o algoritmo EM se mostrou aproximadamente duas ordens de magnitude mais rápido.

No algoritmo iterativo EM, o passo *E* (Esperança) define uma nova função de verossimilhança a cada iteração, a qual é maximizada durante o passo *M* (Maximização). Freqüentemente, os passos *E* e *M* são combinados em um único algoritmo coeso, mas por questões de formalização eles serão divididos em dois passos. Se o passo *M* é utilizado apenas para aumentar a verossimilhança, sem contudo maximizá-la em cada passo, o algoritmo é chamado *Maximização da Esperança Generalizado* (GEM). O algoritmo de

aprendizado denominado máquina de Boltzmann (HAYKIN, 1999), por exemplo, é essencialmente um algoritmo GEM.

A fim de aplicar o algoritmo EM em um novo domínio, um conjunto de variáveis faltantes ou ausentes devem ser consideradas, simplificando assim a otimização do logaritmo da verossimilhança. Haverá distinção, portanto, entre o logaritmo da verossimilhança de dados incompletos $l(\theta;\chi)$ sobre os dados observáveis χ e o logaritmo da verossimilhança de dados completos $l_c(\theta;T)$ sobre os dados completos $T = \chi \cup Z$, que incluem um conjunto Z de variáveis ausentes. É importante notar que o logaritmo da verossimilhança de dados completos é uma variável aleatória, visto que as variáveis ausentes são desconhecidas.

O algoritmo EM tem como objetivo aumentar a estimação do logaritmo da verossimilhança de dados completos. Usando os dados observáveis e o modelo atual, o primeiro passo *E* calcula o valor esperado do logaritmo da verossimilhança de dados completos, na forma:

$$Q(\theta,\theta^{(k)}) = E[l_c(\theta;T) \mid \chi]. \tag{3.43}$$

onde o sub-índice *c* implica que a verossimilhança está sendo avaliada junto aos dados completos. O símbolo sobrescrito *k* refere-se aos parâmetros da *k*-ésima iteração do algoritmo. O passo *E* produz uma função determinística *Q* dos parâmetros do modelo. O passo *M* maximiza a função *Q* com respeito aos parâmetros do modelo, produzindo:

$$\theta^{(k+1)} = \arg\max_{\theta} Q(\theta,\theta^{(k)}). \tag{3.44}$$

Os passos *E* e *M* são repetidos até que o processo de maximização não produza nenhum melhoramento significativo. O algoritmo EM realiza uma estimativa dos parâmetros no sentido de aumentar o valor da função *Q* a cada iteração. A função *Q*, portanto, vai corresponder ao valor esperado do logaritmo da verossimilhança dos dados completos.

Nosso objetivo é maximizar o logaritmo da verossimilhança dos dados incompletos, já que DEMPSTER *et al.* (1977) tratou esta questão e provou que um incremento na função *Q* sempre implica um incremento no logaritmo da verossimilhança de dados incompletos, na forma:

$$Q(\theta,\theta^{(k+1)}) \geq Q(\theta,\theta^{(k)}) \Rightarrow l(\theta^{(k+1)},X) \geq l(\theta^{(k)},X)\,. \tag{3.45}$$

Isto significa que a verossimilhança original aumenta monotonicamente a cada iteração, convergindo para um ótimo local.

3.8.2 Aplicação do algoritmo EM para HMEs

A aplicação do algoritmo EM para a arquitetura HMEs (ver Figura 3.2) envolve a definição de variáveis ausentes que facilitam a otimização do logaritmo da verossimilhança. Seja z_i, i =1,..., N, um conjunto de variáveis indicadoras binárias para a rede *gating* do nível superior, e seja $z_{j|i}$, i,j =1, ...,N, um conjunto de variáveis indicadoras para a rede *gating* da segunda camada (camada inferior). Para qualquer vetor de entrada **x**, exatamente um dos z_i's vale 1, com todos os outros valendo 0. Similarmente, dado z_i, exatamente um dos $z_{j|i}$'s vale 1, com todos os outros valendo 0. Os z_i's e os $z_{j|i}$'s admitem uma interpretação probabilística como decisão (caso o valor de z_i's e $z_{j|i}$'s fossem conhecido) correspondente ao modelo de probabilidade. Uma instanciação dos z_i's e dos $z_{j|i}$'s corresponde a um caminho específico do nó raiz da árvore para uma das folhas, determinando o especialista responsável pela geração dos dados. Observe, portanto, que os z_i's e $z_{j|i}$'s não são conhecidos e devem ser tratados como variáveis randômicas. Caso contrário, o problema de maximização para HMEs deveria ser dividido em um conjunto de problemas de maximização da verossimilhança independentes, para cada rede *gating* e para cada rede especialista. Embora os z_i's e $z_{j|i}$'s sejam desconhecidos, é possível especificar um modelo de probabilidade do logaritmo da verossimilhança de dados completos que os une aos dados observáveis (FRITSH, 1996), possibilitando a aplicação do algoritmo EM, levando a:

$$l_c(\theta;T) = \log \prod_{t=1}^{N} \prod_{i=1}^{r} \prod_{j=1}^{m} g_i^{(t)} g_{j|i}^{(t)} P_{ij}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\theta_{ij})^{z_{ij}^{(t)}}\,, \tag{3.46}$$

$$l_c(\theta;T) = \sum_{t=1}^{N} \sum_{i=1}^{r} \sum_{j=1}^{m} z_{ij}^{(t)} \log\{g_i^{(t)} g_{j|i}^{(t)} P_{ij}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\theta_{ij})\}\,, \tag{3.47}$$

$$l_c(\theta;T) = \sum_{t=1}^{N} \sum_{i=1}^{r} \sum_{j=1}^{m} z_{ij}^{(t)} \{\log g_i^{(t)} + \log g_{j|i}^{(t)} + \log P_{ij}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\theta_{ij})\}\,. \tag{3.48}$$

onde o sub-índice *c* implica que a verossimilhança está sendo avaliada junto aos dados completos $T = \chi \cup Z$. A maximização do logaritmo da verossimilhança dos dados completos é mais fácil que o correspondente logaritmo da verossimilhança dos dados incompletos, pois o logaritmo pode ser trazido para dentro do somatório.

É possível provar que as densidades de probabilidade *a posteriori* h_i, $h_{j|i}$ e h_{ij} podem ser usadas como valores esperados para as variáveis desconhecidas z_i's, $z_{j|i}$'s e z_{ij}'s, respectivamente (JORDAN & JACOBS, 1992). Usando este fato, define-se a função *Q* para o passo *E* do algoritmo EM na forma:

$$Q(\theta,\theta^{(k)}) = \sum_{t=1}^{N}\sum_{i=1}^{r}\sum_{j=1}^{m} h_{ij}^{(t)}\{\log g_i^{(t)} + \log g_{j|i}^{(t)} + \log P_{ij}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})\}. \qquad (3.49)$$

A função *Q* para a arquitetura MEs pode ser derivada de forma similar:

$$Q(\theta,\theta^{(k)}) = \sum_{t=1}^{N}\sum_{i=1}^{r} h_i^{(t)}\{\log g_i^{(t)} + \log P_i(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})\}. \qquad (3.50)$$

O passo *M* requer a maximização da função *Q* com respeito aos modelos dos parâmetros. Nota-se agora o benefício da aplicação do algoritmo EM, já que a maximização divide o problema original em um conjunto de problemas de maximização separáveis, os quais podem ser solucionados independentemente no passo *M*, como segue:

$$\mathbf{v}_i^{k+1} = \arg\max_{v_i} \sum_{t=1}^{N}\sum_{l=1}^{r} h_l^{(t)} \log g_l^{(t)}, \qquad (3.51)$$

$$\mathbf{v}_{j|i}^{k+1} = \arg\max_{v_{j|i}} \sum_{t=1}^{N}\sum_{l=1}^{r} h_l^{(t)} \sum_{k=1}^{m} h_{k|l}^{(t)} \log g_{k|l}^{(t)}, \qquad (3.52)$$

$$\theta_{ij}^{k+1} = \arg\max_{\theta_{ij}} \sum_{t=1}^{N} h_{ij}^{(t)} \log P_{ij}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij}), \qquad (3.53)$$

onde o índice *k* representa a k-ésima iteração do passo M.

É possível obter uma interpretação para as equações acima. Para a rede *gating* do nível superior, deve-se maximizar a entropia cruzada entre a probabilidade *a posteriori* dos ramos h_l e a probabilidade *a priori* dos ramos g_l. Para as redes *gating* de segundo nível, deve-se maximizar a entropia cruzada entre a probabilidade *a posteriori* dos ramos $h_{k|l}$ e a probabilidade *a priori* dos ramos $g_{k|l}$, ponderados pela probabilidade *a posteriori* do próprio nó da rede *gating* h_l. Em árvores profundas, os pesos para a entropia cruzada são

simplesmente o produto da probabilidade *a posteriori* dos ramos ao longo do caminho do nó raiz para o nó da rede *gating* em questão (JORDAN & JACOBS, 1994). Finalmente, o problema de maximização para redes especialistas envolve a maximização da entropia cruzada entre a probabilidade *a posteriori* do especialista e a saída desejada.

O passo *M* do algoritmo EM pode ter diferentes formas, dependendo da arquitetura da rede *gating*, dos especialistas e de qual das variações do algoritmo EM é escolhida. Com respeito à arquitetura da rede, as duas principais opções são perceptrons simples (modelos lineares generalizados) ou perceptrons multicamadas. Com perceptrons simples e densidade condicional exponencial (do qual a densidade gaussiana e *multinomial* são casos especiais), para $P(\mathbf{y} \mid \mathbf{x}, \theta_i)$ na equação (3.38) o problema de otimização reduz-se a problemas de maximização ponderados para modelos lineares generalizados (JORDAN & JACOBS, 1994; MCLACHLAN, 1988) podendo ser solucionados eficientemente com o algoritmo dos quadrados mínimos ponderados iterativamente (IWLS, do inglês *iteratively weighted least-squares*). Uma descrição detalhada do algoritmo IWLS pode ser encontrada em FRITSH (1996) e JORDAN & JACOBS (1994). Para densidade condicional gaussiana, o problema de otimização dos parâmetros da rede especialista reduz-se a um algoritmo de um passo usando pseudo-inversão (MOERLAND, 1997).

Quando a rede especialista e a rede *gating* são escolhidas como perceptrons multicamadas (MLPs), a otimização do gradiente é mais apropriada. Nesta tese, além do método do gradiente, serão empregados o método do gradiente conjugado híbrido (DOS SANTOS & VON ZUBEN, 2000) e o método Levenberg-Marquardt (HAGAN & MENHAJ, 1994). O uso de MLPs tem a vantagem de adicionar maior flexibilidade aos modelos MEs e HMEs. Por outro lado, a convergência para um mínimo local de boa qualidade não pode ser garantida (WEIGEND *et al.*, 1995).

Uma implementação parcial do passo *E* foi proposta em NEAL & HINTON (1993), caracterizando-se basicamente como um algoritmo EM *on-line*. Para todas as variações do algoritmo EM, a convergência em direção a um mínimo local é ainda garantida quando se adotam modelos lineares.

3.9 Mistura de especialistas gaussianos

Até este ponto do trabalho, considerou-se um perceptron simples (modelo linear generalizado) ou um perceptron multicamadas, tanto na rede *gating* como nas redes especialistas de uma mistura hierárquica de especialistas. No entanto, nada impede, em princípio, que se adote uma forma paramétrica arbitrária para a rede *gating* e para as redes especialistas. No caso de classificação, entretanto, os modelos da rede *gating* e das redes especialistas têm que cumprir a restrição de soma unitária de suas saídas, isso para cada vetor de entrada.

XU *et al.* (1995) propuseram usar uma forma paramétrica baseada em modelos gaussianos para a rede *gating*. FRITSH (1996) mostrou que a mesma forma paramétrica pode ser adotada também para as redes especialistas. Tal arquitetura é muito atrativa, pois há mecanismos eficazes para fornecer uma inicialização próxima a uma solução ótima, reduzindo assim o tempo de convergência do algoritmo de aprendizado (FRITSH, 1996). Nas próximas seções, serão apresentados os algoritmos de treinamento e inicialização, e as equações propostas por XU *et al*. (1995) e FRITSH (1996).

3.9.1 Parametrização alternativa

Ao invés de aplicar um modelo linear com não-linearidade *softmax* (JACOBS *et al.*, 1991), a seguinte parametrização foi proposta para a rede *gating* em uma arquitetura de mistura de especialistas:

$$g_i(\mathbf{x},\mathbf{v}) = \frac{\alpha_i P(\mathbf{x} \mid \mathbf{v}_i)}{\sum_{k=1}^{m} \alpha_k P(\mathbf{x} \mid \mathbf{v}_k)}, \text{ com } \sum_{k=1}^{m} \alpha_k = 1 \text{ e } \alpha_k \geq 0 \text{ e}$$

$$P(\mathbf{x} \mid \mathbf{v}_i) = \frac{1}{(2\pi)^{\frac{n}{2}} \left|\Sigma_i\right|^{\frac{1}{2}}} \exp\left\{\frac{1}{2}(\mathbf{x}-\gamma_i)^T \Sigma_i^{-1} (\mathbf{x}-\gamma_i)\right\}. \tag{3.54}$$

onde n é a dimensão da entrada $\mathbf{x}$.

Esta formulação é válida desde que os coeficientes g_i (i=1,...,m) produzam soma unitária para todo vetor de entrada $\mathbf{x}$. A forma paramétrica acima pode ser interpretada como um classificador paramétrico *a posteriori*, de acordo com o teorema de Bayes:

$$p(\mathbf{w}_i \mid \mathbf{x}) = \frac{P(\mathbf{w}_i)\, p(\mathbf{x} \mid \mathbf{w}_i)}{\sum_{k=1}^{m} P(\mathbf{w}_k)\, p(\mathbf{x} \mid \mathbf{w}_k)}, \tag{3.55}$$

onde a probabilidade *a priori* $P(\mathbf{w}_i)$ é dada por α_i (i=1,...,m) e a classe de probabilidade $p(\mathbf{x} \mid \mathbf{w}_i)$ é modelada por uma única distribuição gaussiana.

3.9.2 Classificador gaussiano para a rede *gating*

A parametrização da rede *gating* de uma mistura de especialistas na forma de um classificador gaussiano *a posteriori* permite derivar um eficiente algoritmo de um único passo EM para estimar os parâmetros da rede *gating*. Além disso, a forma paramétrica especial permite inicializar as funções gaussianas e a probabilidade *a priori* de forma a aumentar significativamente a velocidade do treinamento.

3.9.2.1 O algoritmo EM

A probabilidade condicional fundamental em uma mistura de especialistas, conforme já mencionado, é:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_{i=1}^{m} g_i(\mathbf{x}, \mathbf{v}_i) P(\mathbf{y} \mid \mathbf{x}, \theta_i), \tag{3.56}$$

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_{i=1}^{m} \frac{\alpha_i P(\mathbf{x} \mid \mathbf{v}_i)}{\sum_{k=1}^{m} \alpha_k P(\mathbf{x} \mid \mathbf{v}_k)} P(\mathbf{y} \mid \mathbf{x}, \theta_i), \tag{3.57}$$

Caso a intenção seja derivar um algoritmo EM diretamente sobre esta mistura de densidades, será constatado que o passo *M* não é analiticamente solucionável e deveria requerer processamento iterativo similar ao algoritmo IRLS (*iteratively recursive least-squares,* na literatura em inglês). Entretanto, a mistura condicional acima pode ser reescrita em uma forma que permite uma solução analítica para o problema de maximização da verossimilhança (FRITSH, 1996), a saber:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_{i=1}^{m} \alpha_i P(\mathbf{x} \mid \mathbf{v}_i) P(\mathbf{y} \mid \mathbf{x}, \theta_i). \tag{3.58}$$

Ao invés de estimar os parâmetros da rede *gating* para maximizar a verossimilhança da densidade de mistura original, é possível maximizar a verossimilhança da densidade conjunta, que conduz ao seguinte logaritmo da verossimilhança:

$$l(\theta, T) = \sum_{t=1}^{N} \log \sum_{j=1}^{m} \alpha_j P(\mathbf{x}^{(t)} \mid \mathbf{v}_j) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i) . \tag{3.59}$$

Na estrutura EM, a função de verossimilhança completa assume a forma:

$$l(\theta, T) = \sum_{t=1}^{N} \sum_{j=1}^{m} z_j^{(t)} \log\left(\alpha_j P(\mathbf{x}^{(t)} \mid \mathbf{v}_j) P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i)\right), \tag{3.60}$$

onde o valor esperado da variável ausente, usando a regra de Bayes, produz:

$$\mathrm{E}(\mathbf{z}_j^{(t)}) = P(\mathbf{z}_j^{(t)} = 1 \mid \mathbf{y}^{(t)}, \mathbf{x}^{(t)}) = \frac{P(\mathbf{y}^{(t)} \mid \mathbf{z}_j^{(t)} = 1, \mathbf{x}^{(t)}) P(\mathbf{z}_j^{(t)} = 1 \mid \mathbf{x}^{(t)})}{p(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})} . \tag{3.61}$$

Usando a interpretação probabilística de MEs, resulta:

$$\mathrm{E}(z_j^{(t)}) = \frac{\alpha_j P(\mathbf{x}^{(t)}) \phi_j(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})}{\sum_{i=1}^{m} \alpha_i P_i(\mathbf{x}^{(t)}) \phi_j(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})} = h_j(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) . \tag{3.62}$$

A esperança da verossimilhança completa assume a forma:

$$Q(\theta^{k+1}, \theta) = \sum_{t=1}^{N} \sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \log\left(\alpha_j P(\mathbf{x}^{(t)} \mid \mathbf{v}_j)\right) + \sum_{t=1}^{N} \sum_{j=1}^{m} h(\mathbf{x}^{(t)}, \mathbf{y}^{(t)}) \log\left(P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_i)\right) \tag{3.63}$$

Comparando a equação (3.50) com a equação (3.63), observa-se que a função-objetivo do especialista (segunda parte da equação) não se altera e que, conseqüentemente, o passo *M* para as redes especialistas não é modificado.

Analisando a equação (3.50), pode-se observar que o primeiro termo depende somente dos parâmetros da rede *gating*, enquanto o segundo termo depende somente dos parâmetros das redes especialistas. Logo, a equação (3.50) pode ser decomposta em:

$$Q(\theta^{k+1}, \theta) = Q^g(\theta^{k+1}, \theta) + Q^e(\theta^{k+1}, \theta)$$

onde $Q^g(\theta^{k+1}, \theta)$ é a função de verossimilhança da rede gating e corresponde ao primeiro termo da equação (3.50), e $Q^e(\theta^{k+1}, \theta)$ é a função de verossimilhança da rede especialistas e corresponde ao segundo termo da equação (3.50).

Então os parâmetros da rede *gating* podem ser obtidos tomando as derivadas parciais da verossimilhança da rede *gating*, a saber:

$$Q^g(\theta^{k+1},\theta) = \sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log(\alpha_j P(\mathbf{x}^{(t)} \mid \mathbf{v}_j)), \tag{3.64}$$

$$Q^g(\theta^{k+1},\theta) = \sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log(\alpha_j) + \sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log\left(P(\mathbf{x}^{(t)} \mid \mathbf{v}_j)\right). \tag{3.65}$$

Esta é exatamente a função-objetivo que é maximizada quando aplicamos o algoritmo EM para um simples modelo de mistura gaussiana (veja, por exemplo, seção 2.6 em BISHOP (1995)). Analisando a equação (3.65) pode-se observar que o primeiro termo depende somente de α_j e o segundo termo somente de $P(\mathbf{x}^{(t)} \mid \mathbf{v}_j)$. Logo, a equação (3.65) pode ser decomposta em dois termos, sendo o primeiro definido por:

$$\sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log(\alpha_j) \tag{3.66}$$

sujeito a: $\sum_{j=1}^{m} \alpha_j = 1$

e o segundo termo definido por:

$$\sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log\left(P(\mathbf{x}^{(t)} \mid \mathbf{v}_j)\right) \tag{3.67}$$

Portanto, a maximização da função-objetivo da rede *gating* pode ser realizada maximizando as equações (3.66) e (3.67) independentemente. Para maximizar a equação (3.66) pode-se aplicar a técnica de Lagrange, a qual conduz à seguinte função lagrangeana:

$$L(\alpha,\lambda) = -\left(\sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}) \log \alpha_j + \lambda\left(\sum_{j=1}^{m}\alpha_j - 1\right)\right), \tag{3.68}$$

onde λ é o multiplicador de lagrange.

Tomando a derivada parcial com respeito a α e λ, obtém-se um sistema de m+1 equações lineares, cuja solução assume a seguinte forma:

$$\alpha_j^{(k+1)} = \tfrac{1}{N}\sum_{t=1}^{N} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)}), \tag{3.69}$$

onde N é o número de padrões no conjunto de treinamento e o índice (k+1) representa a estimação de α_j no (k+1)-ésimo passo M.

Os centros das funções gaussianas pode ser obtidos maximizando a segunda parte da função-objetivo definido em (3.67). Calculando a derivada da equação (3.67) em relação aos centros γ_j, obtém-se :

$$\frac{\partial\left(-\sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)})\log(P_j(\mathbf{x}^{(t)}))\right)}{\partial\gamma_j} = -\sum_{t=1}^{N}\frac{h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)})}{P_j(\mathbf{x})}\frac{\partial P_j(\mathbf{x}^{(t)})}{\partial\gamma_j} = -\sum_{t=1}^{N} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)})\frac{(\mathbf{x}^{(t)}-\gamma_j)}{\sigma_j^2}. \tag{3.70}$$

Fazendo esta derivada igual a zero, obtém-se uma nova estimativa para a média, isto é, os centro das funções gaussianas, dadas por:

$$\gamma_j = \frac{\sum_{t=1}^{N} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)})\mathbf{x}^{(t)}}{\sum_{t=1}^{N} h_j^{(t)}(\mathbf{x}^{(t)},\mathbf{y}^{(t)})}. \tag{3.71}$$

Para a matriz de covariância local das funções gaussianas, calculando-se a derivada parcial da segunda parte da função de verossimilhança da rede *gating* e fazendo-se esta derivada igual a zero, a nova estimativa da variância local é expressa como segue:

$$\Sigma_i^{(k+1)} = \frac{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})\left[\mathbf{x}^{(t)} - \gamma_i^{(k+1)}\right]\left[\mathbf{x}^{(t)} - \gamma_i^{(k+1)}\right]^T}{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})} \tag{3.72}$$

Com isso, está completo o passo *M* para funções gaussianas. É importante observar que, embora se esteja verdadeiramente otimizando a probabilidade conjunta durante o treinamento, o desenvolvimento ainda está baseado no modelo de mistura condicional (FRITSH, 1996). Outra possível extensão é o uso de funções exponenciais (XU *et al.*, 1995). No entanto, esta extensão foge ao escopo desta tese.

Logo, o algoritmo EM conduz ao seguinte método de estimação iterativo:

Passo *E* – Para cada vetor de treinamento, calcule a probabilidade *a posteriori* h_i de acordo com:

$$h_i^{(k)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}) = \frac{\alpha_i^{(k)} P(\mathbf{x}^{(t)} \mid \mathbf{v}_i^{(t)}) P_i(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \Theta_{(i)}^{(k)})}{\sum_{j=1}^{m} \alpha_j^{(k)} P(\mathbf{x}^{(t)} \mid \mathbf{v}_j^{(t)}) P_j(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \Theta_{(j)}^{(k)})} \tag{3.73}$$

Passo *M* – Use os h_i*'s* para calcular as novas estimativas para os parâmetros α_i, γ_i e Σ_i da rede *gating*. A nova estimativa pode ser computada diretamente, pois o problema de maximização da verossimilhança é agora analiticamente solucionável, produzindo:

$$\alpha_i^{(k+1)} = \frac{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})}{\sum_{t=1}^{N}\sum_{k=1}^{m} h_k^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})}, \tag{3.74}$$

$$\gamma_i^{(k+1)} = \frac{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})\mathbf{x}^{(t)}}{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})}, \tag{3.75}$$

$$\Sigma_i^{(k+1)} = \frac{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})\left[\mathbf{x}^{(t)} - \gamma_i^{(k+1)}\right]\left[\mathbf{x}^{(t)} - \gamma_i^{(k+1)}\right]^T}{\sum_{t=1}^{N} h_i^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)})}. \tag{3.76}$$

O problema de maximização da verossimilhança para as redes especialistas permanece sem solução analítica (no caso de classificação) e seus parâmetros devem ser estimados iterativamente pelo gradiente descendente ou por quadrados mínimos recursivos. Portanto, o algoritmo EM descrito acima para a rede *gating* é computacionalmente mais eficiente que o algoritmo IRLS para modelos lineares generalizados (GLIMs). Observe que o cálculo da probabilidade *a posteriori* do nó h_i é diferente para os algoritmos EM e GLIMs. Esta diferença influencia indiretamente a estimação dos parâmetros das redes especialistas, pois a probabilidade *a posteriori* conjunta do nó aparece nas fórmulas de re-estimação das redes especialistas.

Observe também que a formulação do algoritmo EM apresentada acima maximiza a soma da mistura de verossimilhança e a verossimilhança condicional da rede *gating*, ao invés de maximizar a própria mistura de verossimilhança. Durante a fase de teste, portanto, a saída da mistura ainda segue o modelo para mistura de HMEs.

3.9.2.2 Inicialização

A forma paramétrica que foi aplicada à rede *gating* é mais atrativa porque permite a inicialização dos parâmetros próximo ao valor ótimo. Há vários trabalhos importantes sobre inicialização de modelos de mistura gaussiana e redes com funções de base radial que podem ser adotados aqui (WATERHOUSE, 1998; FRITSH, 1996). De fato, desde que se

admita que a forma paramétrica pode ser vista como um classificador *a posteriori*, seus parâmetros podem ser melhor inicializados estimando-se a probabilidade *a priori* e a verossimilhança da classe pela freqüência relativa, e maximizando a estimação da verossimilhança. Entretanto, no caso de uma rede *gating* em uma mistura de especialistas, não existem rótulos das classes para estimar os parâmetros de um classificador gaussiano. Uma alternativa foi proposta em FRITSCH (1996), a qual será empregada mais adiante, quando serão adotados classificadores gaussianos também para as redes especialistas.

Uma possível técnica de inicialização para redes *gating* gaussianas, capaz de operar adequadamente na prática, é estimar os parâmetros tais que a verossimilhança dos dados dentro de um modelo de mistura não-supervisionado seja maximizado. Com isso, a inicialização dos parâmetros da rede *gating* é dada na forma:

$$\hat{\mathbf{v}}_i = \arg\max_{v_i} \sum_{t=1}^{N} \log \sum_{i=1}^{m} \sum_{i} \alpha_i^{(t)} P(\mathbf{x}^{(t)}, \mathbf{v}_i), \tag{3.77}$$

com $\sum_{k=1}^{m} \alpha_k = 1$ e $\alpha_k \geq 0$. Usualmente, a maximização da verossimilhança é feita nos três passos a seguir:

a) **Extração de amostras**: Inicializar a média da gaussiana, extraindo randomicamente um número apropriado de amostras do conjunto de treinamento.
b) **Média dos clusters**: Aplicar um algoritmo de clusterização, como o algoritmo de *k* - means. Inicializar cada γ_i como a média de cada cluster.
c) **Maximização da verossimilhança**: Iterativamente, deve-se estimar novamente os coeficientes α_i, a média γ_i e as matrizes de covariância Σ_i de acordo com o algoritmo EM para mistura de gaussianas (DEMPSTER *et al.*, 1977).

A possibilidade de inicializar os parâmetros da rede *gating* próximo à solução ótima e a existência de um algoritmo de re-estimação em um único laço EM conferem à parametrização gaussiana uma poderosa extensão para arquiteturas HMEs.

3.10 Modelos gaussianos para os especialistas

Dadas às vantagens da parametrização gaussiana da rede *gating*, seria desejável poder empregar a mesma parametrização para as redes especialistas. Infelizmente, a solução para o problema de aprendizado EM proposto em XU *et al.* (1995) não pode ser generalizado para as redes especialistas. A restrição do algoritmo EM irá, então, ser relaxada e será derivado um algoritmo EM generalizado, que somente garante aumentar a verossimilhança da mistura a cada iteração, ao invés de garantir maximizá-la.

3.10.1 Algoritmo GEM

Como mencionado, será agora derivado um algoritmo EM generalizado para mistura de especialistas que emprega parametrização gaussiana tanto para a rede *gating* como para as redes especialistas, proposto em FRITSCH (1996). O modelo de probabilidade da arquitetura global é dado por:

$$P(\mathbf{y} \mid \mathbf{x}, \Theta) = \sum_{i=1}^{m} g(\mathbf{x}, \mathbf{v}_i) P_i(\mathbf{y} \mid \mathbf{x}, \theta_i), \tag{3.78}$$

onde P_i (i=1,...,m) são densidades *multinomiais*, modelando a tarefa de classificação de múltiplos modos imposta aos especialistas, v_i e θ_i formam o conjunto de parâmetros para a rede *gating* e as redes especialistas, respectivamente. A ativação de cada especialista é calculada da mesma forma que a ativação da rede *gating*, tomando um classificador gaussiano *a posteriori*, na forma:

$$\mu_{ij}(\mathbf{x}, \theta_i) = \frac{\alpha_{ij} P(\mathbf{x} \mid \theta_{ij})}{\sum_{k=1}^{K} \alpha_{ik} P(\mathbf{x} \mid \theta_{ik})} \text{ com } \sum_{k=1}^{K} \alpha_{ik} = 1 \text{ e } \alpha_{ik} \geq 0, \tag{3.79}$$

$$P(\mathbf{x} \mid \theta_{ij}) = \frac{1}{(2\pi)^{\frac{n}{2}} \left|\Sigma_{ij}\right|^{\frac{1}{2}}} \exp\left\{\frac{1}{2}(\mathbf{x} - \mu_{ij})^T \Sigma_{ij}^{-1} (\mathbf{x} - \mu_{ij})\right\}. \tag{3.80}$$

onde K é o numero de classes e n é a dimensão do espaço de entrada.

A ativação do especialista pode ser re-escrita em uma forma interessante:

$$\mu_{ij}(\mathbf{x}, \theta_i) = \frac{\exp(u_{ij})}{\sum_{k=1}^{K} \exp(u_{ik})}, \tag{3.81}$$

com $u_{ij} = \log(\alpha_{ij}) - \frac{1}{2}\left[n\log(2\pi) + \log\left|\Sigma_{ij}\right| + (\mathbf{x}-\mu_{ij})^T \Sigma_{ij}^{-1}(\mathbf{x}-\mu_{ij})\right]$.

A ativação do especialista foi expressa usando a mesma não-linearidade *softmax* adotada no caso do GLIM (*Generalized Linear Model*). A diferença está na troca de um modelo linear básico que calcula $u_{ij} = \theta \cdot X$ por um modelo radial, que basicamente computa $u_{ij} = (X-\theta)^2$. Expressando o novo modelo em termos da função *softmax*, é possível unificar os modelos radiais para redes especialistas.

O passo *M* de um algoritmo EM para mistura de especialistas envolve a maximização das duas verossimilhanças fornecidas a seguir (supondo um modelo de probabilidade *multinomial*):

$$v_i^{(k+1)} = \arg\max_{\mathbf{v}_i} \sum_{t=1}^{N}\sum_{j=1}^{m} h_j^{(t)} \log g_j^{(t)}, \tag{3.82}$$

$$\theta_i^{(k+1)} = \arg\max_{\theta_i} \sum_{t=1}^{N} h_i^{(t)} \sum_{j=1}^{m} \mathbf{y}_j^{(t)} \log \mu_{ij}^{(t)}, \tag{3.83}$$

onde $\mathbf{y}_j^{(t)}$ é a saída desejada para o nó de saída das redes especialistas. Por causa da não-linearidade da função *softmax* em ambos *g* e μ, não há forma fechada para a solução deste problema. Portanto, recorre-se à derivação de um algoritmo GEM que aumenta a verossimilhança usando o gradiente ascendente, na forma:

$$\mathbf{v}_i^{(k+1)} = \mathbf{v}_i^{(k)} + \rho \sum_{t=1}^{N}\left[\sum_{j=1}^{m} h_j^{(t)}(\delta_{ij} - g_j^{(t)})\right]\frac{\partial u_i}{\partial \mathbf{v}_i}, \tag{3.84}$$

$$\theta_{ij}^{(k+1)} = \theta_{ij}^{(k)} + \rho \sum_{t=1}^{N} h_i^{(t)}\left[\sum_{l=1}^{K} \mathbf{y}_j^{(t)}(\delta_{jl} - \mathbf{y}_{il}^{(t)})\right]\frac{\partial u_j}{\partial \theta_{ij}}, \tag{3.85}$$

onde δ_{ij} é o delta de Kroneker, ρ é a taxa de aprendizado e u_i (i=1,...,m) são as funções radiais ou lineares *a priori* para a não-linearidade *softmax*. No caso de especialistas gaussianos com matrizes de covariância diagonal, obtém-se a seguinte regra de atualização para os parâmetros de um especialista específico *i* na MEs:

$$\alpha_j^{(t+1)} = \alpha_j^{(t)} + \eta \sum_{t=1}^{N} h_i^{(t)}\left[\sum_{l=1}^{K} \mathbf{y}_l^{(t)}(\delta_{jl} - \mathbf{y}_l^{(t)})\right]\frac{1}{\alpha_j^{(t)}}, \tag{3.86}$$

$$\mu_j^{(t+1)} = \mu_j^{(t)} + \eta \sum_{t=1}^{N} h_i^{(t)}\left[\sum_{l=1}^{K} \mathbf{y}_l^{(t)}(\delta_{jl} - \mathbf{y}_l^{(t)})\right]\frac{(\mathbf{x}_m - \mu_{jm}^{(t)})}{\alpha_j^{2(t)}}, \tag{3.87}$$

$$\sigma_{jm}^{2(t+1)} = \sigma_{jm}^{2(t)} + \eta \sum_{t=1}^{N} h_i^{(t)} \left[\sum_{l=1}^{K} \mathbf{y}_l^{(t)} (\delta_{jl} - \mathbf{y}_l^{(t)}) \right] \frac{(\mathbf{x}_m - \mu_{jm}^{(t)})^2 - \sigma_{jm}^{2(t)}}{\sigma_{jm}^{4(t)}}. \qquad (3.88)$$

onde η é a taxa de aprendizado.

Os α_j's necessitam ser normalizados depois de cada iteração, a fim cumprir a restrição de que sua soma produza valor unitário. Para aumentar a velocidade de convergência, é possível usar este algoritmo na versão baseada no gradiente estocástico, atualizando os parâmetros cada vez que *n* amostras de treinamento são apresentadas. O algoritmo GEM descrito acima é basicamente uma técnica de primeira ordem. Portanto, pode-se argumentar que a velocidade de convergência pode ser muito lenta para conferir utilidade a este algoritmo. Entretanto, FRITSCH (1996) mostrou que a combinação deste algoritmo com a técnica de inicialização apresentada na seção 3.9.2.2 produz uma convergência muito rápida, na prática.

3.11 Variância adaptativa na mistura de especialistas

As seções anteriores focalizaram MEs ou HMEs como um único estimador para a predição da média condicional dos dados desejados. Esta abordagem é muito conveniente para dados que possam ser descritos com a média dependente da entrada e um parâmetro de variância global (MOERLAND, 1997). Portanto, para melhorar a modelagem dos dados é freqüentemente útil ter informações adicionais a respeito dos dados.

Uma possibilidade que tem sido explorada é a estimação da barra de erro local[4] para regressão não-linear. Isto fornece uma estimativa da confiabilidade da predição e a possibilidade de levar em conta ruídos dependentes da entrada. Quanto à estrutura da maximização da verossimilhança, tem sido proposta uma única função de densidade de probabilidade condicional isotrópica (mesmas propriedades em todas as direções) (NIX & WEIGEND,1995), e também a generalização da matriz de covariância arbitrária (WILLIAMS, 1996). Uma desvantagem bem conhecida da estimação do máximo da verossimilhança, entretanto, é que suas estimativas são polarizadas e conduzem à subestimação da variância

[4] É uma linha que se estende do valor mínimo da grandeza ao valor máximo, dando o tamanho dos erros.

do ruído e sobre-ajuste aos dados de treinamento. Para evitar estes problemas, técnicas bayesianas foram aplicadas (BISHOP & QAZAZ, 1997).

Para Mês, é usual introduzir uma variância local para cada especialista (WEIGEND, 1995) dada por:

$$P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij}) = \frac{1}{(2\pi\sigma_j^2)^{\frac{d}{2}}} \exp\left\{ - \frac{\left\| \mathbf{y}^{(t)} - \mu_{ij}^{(t)}(\mathbf{x}) \right\|}{2\sigma_j^2} \right\} \tag{3. 89}$$

onde *d* a dimensão do espaço de saída.

O efeito desta mudança é que o modelo passa a poder lidar com níveis de ruídos diferentes, muito útil ao tratar séries temporais estacionárias por partes, as quais se alternam entre diferentes regimes. WEIGEND *et al.* (1995) mostra que ela reduz o sobre-ajuste e facilita a subdivisão do problema entre os especialistas. A introdução da variância dos especialistas causa uma pequena mudança em várias equações do treinamento do especialista, apresentadas nas seções anteriores. A mudança nos pesos da rede especialista (*i,j*) torna-se proporcional a:

$$\frac{\partial Q^e(\theta^{k+1}, \theta)}{\partial \mu_{ijk}} = h_{ij}^{(t)} \frac{1}{\sigma_j^2} (\mathbf{y}_k^{(t)} - \mu_{ijk}) \tag{3.90}$$

O fator adicional $1/\sigma_j^2$ pode ser visto como uma forma de regressão ponderada na qual o foco é sobre regiões de baixo ruído e que diminui em regiões de alto ruído (*outliers*, por exemplo). A atualização para a variância é facilmente obtida calculando a derivada parcial $\partial Q^e(\theta^{k+1}, \theta)/\partial \sigma_j$, usando a definição $P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_{ij})$ da equação (3.89):

$$\sum_{t=1}^{N} \frac{\partial Q^e}{\partial \sigma_j} = \sum_{t=1}^{N} \left(- \frac{g_j^{(t)}}{\sum_{i=1}^{m} g_j^{(t)} P_{ij}^{(t)}} \frac{\partial P_{ij}^{(t)}}{\partial \sigma_j} \right) = \sum_{t=1}^{N} \left(- \frac{g_j^{(t)}}{\sum_{i=1}^{m} g_j^{(t)} P_{ij}^{(t)}} \left[P_{ij}^{(t)} \frac{\left\| y^{(t)} - \mu_j^{(t)} \right\|^2}{\sigma_j^3} \right] - \frac{dP_{ij}^{(t)}}{\sigma_j} \right) \tag{3.91}$$

Usando a definição de h_{ij} e fazendo a derivada parcial igual a zero, uma solução direta (para atualização em lote) é:

$$\sigma_j^2 = \frac{1}{d} \frac{\sum_{t=1}^{N} h_j^{(t)} \left\| \mathbf{y}^{(t)} - \mu_j^{(t)} \right\|^2}{\sum_{t=1}^{N} h_j^{(t)}(\mathbf{x}^{(t)}, \mathbf{y}^{(t)})} \tag{3.92}$$

Logo a variância σ_j^2 do especialista *j* será a média ponderada do erro quadrático dividido pela soma dos pesos, onde os pesos são a probabilidade *a posteriori*. WEIGEND *et al.* (1995) também descreveu a incorporação de crença *a priori* a respeito da variância do especialista na estrutura de maximização da verossimilhança. A fim de evitar estimativas polarizadas e sobre-ajuste, a abordagem bayesiana foi também aplicada para MEs (WATERHOUSE *et al.*, 1996).

A estimação da barra de erro local da variância dos especialistas é direta (seção 6.4 de BISHOP, 1995) usando a definição de MEs da equação (3.4), ou de HME da equação (3.16), e de P_j incluindo variância dos especialistas, na equação (3.89), produzindo:

$$\sigma(\mathbf{x}) = \sum_{j=1}^{m} g_j(\mathbf{x})(\sigma_j^2 + \| \mathbf{y}(\mathbf{x}) - \mu_j(\mathbf{x}) \|^2) . \qquad (3.93)$$

De fato, BISHOP (1995) segue uma abordagem mais geral, onde a variância dos especialistas é dependente da entrada, e que permite uma modelagem da distribuição de probabilidade condicional.

3.12 Melhorando o desempenho de generalização das redes especialistas

Devido ao tamanho finito do conjunto de treinamento, é muito difícil determinar o conjunto ótimo de parâmetros para modelar um dado problema. Redes neurais são normalmente treinadas com conjuntos de dados de treinamento, sendo que um dos grandes problemas com a quantidade finita dos dados é que o treinamento excessivo da rede neural junto aos dados de treinamento pode, caso estes não sejam realmente representativos, conduzir a um fenômeno conhecido como sobre-ajuste (*overfitting*): a rede começa a aprender peculiaridades e a perder a capacidade de generalizar corretamente frente a dados diferentes oriundos da mesma distribuição.

A fim de não sobre-ajustar aos dados de treinamento, uma prática comum no treinamento de regressores ou classificadores é impor uma restrição de penalidade para penalizar ajustes indesejáveis e promover ajustes suaves (WEIGEND, 1991; BISHOP, 1991).

RAMAMURTIR & GHOSH (1997) introduziram uma restrição de penalidade que atinge a rede *gating* (com kernels gaussianos) e (indiretamente) melhora o desempenho de generalização da mistura de redes especialistas.

No treinamento da arquitetura de mistura de especialistas, a rede *gating* divide o espaço de entrada em sub-regiões e a rede especialista trabalha sobre estas sub-regiões individuais. Os dados de treinamento são freqüentemente não-uniformemente distribuídos ao longo dos valores de entrada e pode haver mais dados em alguma região que em outras. O treinamento é normalmente realizado dividindo os dados disponíveis em conjunto de treinamento e conjunto de validação, treinando a rede sobre o conjunto de dados de treinamento e monitorando o desempenho da rede sobre o conjunto de validação. Quando nenhuma melhora significativa sobre o conjunto de validação é observado, o treinamento é parado. Na arquitetura de mistura de especialistas, durante o progresso do treinamento, é possível que alguns especialistas sejam treinados mais rápido que outros. Como somente o desempenho do conjunto de validação global é monitorado a todo instante, poderá acontecer que alguns especialistas estejam já na fase de sobre-ajuste enquanto o desempenho da mistura sobre o conjunto de validação está ainda melhorando. O sobre-ajuste é provável para um dado especialista se a quantidade de dados for baixa na região sobre a qual ela age. Uma forma de aliviar o problema de sobre-ajuste nas redes especialistas é aumentar a quantidade efetiva de dados vistos pela rede. Isto pode ser obtido caso a rede *gating* separe suavemente os dados de entrada.

A fim de fazer a rede *gating* produzir separações suaves, RAMAMURTIR & GHOSH (1997) definiram um termo de penalidade para ser minimizado juntamente com a maximização do logaritmo da verossimilhança no treinamento da rede especialista:

$$Custo = Q(\theta^{k+1}, \theta) - \lambda * P_e, \tag{3.94}$$

onde λ é um parâmetro de suavização e a penalidade P_e é da forma:

$$P_e = \sum_{t=1}^{N} \sum_{j=1}^{m} (g_j^{(t)})^2 . \tag{3.95}$$

Os índices t e j correspondem aos padrões de treinamento e ao especialista, respectivamente, e $g_j^{(t)}$ é a saída da rede *gating* correspondente ao especialista j e ao padrão

de treinamento *t*. Como já temos $\sum_{j=1}^{m} g_j^{(t)} = 1$, tal termo de penalidade deverá ajudar a rede *gating* a ser eventualmente melhor distribuída sobre os especialistas.

A fim de reduzir a penalidade durante o treinamento, usa-se o gradiente ascendente, diferenciando o termo de penalidade acima para média e variância da rede *gating*. Os α's da rede *gating* com *kernels* normalizados não são considerados no aprendizado do gradiente ascendente, pois é difícil assegurar que os α's somem 1 e sejam todos positivos. A expressão do gradiente ascendente em lote correspondente ao termo de penalidade acima é a seguinte:

$$\Delta\gamma_{ik} = -\lambda \sum_{t=1}^{N} g_i^{(t)} (g_i^{(t)} - \sum_{j=1}^{m} (g_j^{(t)})^2) \frac{(\mathbf{x}_k^{(t)} - \gamma_{ik})}{\sigma_{ik}^2}, \tag{3.96}$$

$$\Delta\sigma_{ik} = -\lambda \sum_{t=1}^{N} g_i^{(t)} (g_i^{(t)} - \sum_{j=1}^{m} (g_j^{(t)})^2) \frac{(\mathbf{x}_k^{(t)} - \gamma_{ik})^2 - \sigma_{ik}^2}{\sigma_{ik}^3}. \tag{3.97}$$

O índice *k* corresponde ao *k*-ésimo componente do vetor de entrada. O termo $\sum_{t=1}^{N} g_i^{(t)} (g_i^{(t)} - \sum_{j=1}^{m} (g_j^{(t)})^2)$ em cada uma das regras de atualização acima diz que a penalidade é zero quando $g_i^{(t)} = \sum_{j=1}^{N} (g_j^{(t)})^2$ o que ocorre quando todos os especialistas são igualmente ponderados. λ é um parâmetro de suavização que, quando alto, favorece ajustes suaves (uma grande variação de *g*) e, quando baixo, favorece ajustes abruptos. As atualizações acima são adicionadas às equações (3.75) e (3.76) enquanto o treinamento é realizado.

3.13 Extensões adotadas em implementações práticas

Nesta seção, serão apresentadas algumas extensões propostas na literatura visando ganho de desempenho, com boa parte delas sendo adotada na obtenção dos resultados práticos desta pesquisa.

3.13.1 Variação nas iterações do passo M

Embora o algoritmo básico imponha que o passo M seja repetido até haver convergência, o algoritmo EM generalizado (GEM) relaxa esta restrição, requerendo somente um aumento na verossimilhança no passo M. Reduzindo o número de iterações do passo M, a computação global também é reduzida. O poder do algoritmo EM reside no passo E, que repetidamente calcula os novos parâmetros baseados no passo M anterior (WATERHOUSE & ROBINSON, 1994).

3.13.2 Taxa de aprendizado

O algoritmo IRLS, assim como o algoritmo de gradiente ascendente, é sensível à taxa de aprendizado. Taxas de aprendizado muito grandes podem causar instabilidade. WATERHOUSE (1998) propôs uma taxa de aprendizado de 0,4 para os especialistas e a rede *gating*, como aquela capaz de fornecer uma boa harmonia entre velocidade de aprendizado e estabilidade, embora uma taxa de 0,8 venha sendo empregada para algumas condições iniciais em modelos lineares, com manutenção de estabilidade.

3.13.3 Saturação do especialista e da saída da rede *gating*

Se a saída $\mu_{ij}^{(t)}$ de qualquer rede especialista torna-se próxima de 1 ou 0, ou se os pesos $h_{ij}^{(t)}$ são próximos de zero, então a adição à matriz hessiana (nos métodos de segunda ordem) da saída *i* daquela rede no tempo *t* será próxima de zero. Se isto ocorrer para a grande maioria do conjunto de treinamento, a hessiana se tornará quase singular, conduzindo a uma perda de precisão numérica em sua inversão. A solução para este problema, mencionada por (WATERHOUSE & ROBINSON, 1994), foi usar um valor limiar para a saída de 0,9999 e 0,0001, e um valor mínimo para os pesos $h_{ij}^{(t)}$ de 0,0001.

3.13.4 Escolha dos valores dos parâmetros iniciais

Existem duas estratégias para inicializar os parâmetros tanto da rede *gating* quanto das redes especialistas. A primeira consiste em começar com todos os vetores de parâmetros do especialista e da rede *gating* iguais a zero e atribuir a uma das saída da rede *gating* valor 1 e para o resto valor 0, de forma que a região de atuação de cada especialista seja distinta, isto é, não haja sobreposição e que cada especialista produza saídas diferentes. A segunda é inicializar todos os vetores de parâmetros com valores aleatórios, no intervalo [–r, r]. Tipicamente, $0{,}1 \leq r \leq 3$ (FRITSH, 1996).

Uma terceira abordagem é usar valores aleatórios para os parâmetros dos especialistas e zeros para os parâmetros iniciais da rede *gating*, com a escolha da estratégia variando de acordo com o problema. WATERHOUSE & ROBINSON (1994) usaram em seus experimentos a segunda opção, porque esta conduziu a resultados mais rápidos e que são menos sensíveis a máximos locais, enquanto a primeira e a terceira opção produziram soluções que caíram em máximos locais ou falharam em separar os especialistas.

3.13.5 Evitando singularidades

Vários resultados práticos surgem na implementação de especialistas para regressão. O mais notável é o comportamento do algoritmo em regiões de baixo ruído no conjunto de dados. Considere uma região na qual um especialista tem baixa contribuição para a predição total (isto é, $\sum_{t=1}^{N} h_{ij}^{(t)}$ é pequeno) e o erro deste especialista é também pequeno (isto é, $\sum_{t=1}^{N} h_{ij}^{(t)} (\mathbf{y}^{(t)} - \mu_{ij}^{(t)})^2$ é pequeno). Isto implica que a variância estimada tende a uma singularidade fazendo surgir uma verossimilhança ilimitada. Este problema é também encontrado em outros modelos de mistura, tais como mistura de gaussianas, como observado por muitos autores, incluindo GELMAN *et al.* (1995) e TITTERINGTON *et al.* (1985). Em geral, uma abordagem simples é atribuir um limite mínimo à variância, baseado em algum conhecimento *a priori*. Esta abordagem foi usada por WEIGEND *et al.* (1995).

WATERHOUSE (1998) impôs uma condição *a priori* sobre o parâmetro da variância, e maximizou a distribuição *a posteriori*, ao invés da verossimilhança. Isto foi feito puramente com propósito computacional, assim como para evitar a singularidade na solução da variância surgida da maximização da verossimilhança. Outros procedimentos podem ser adotados, tais como a não atualização da variância quando o número de observações para um especialista cai abaixo de um limiar, ou então a limitação da variância a um nível específico.

MACKAY (1997) propôs uma distribuição gama *a priori* sobre a recíproca da variância $\beta_i = 1/\sigma_i^2$ para modelos de regressão linear e não-linear. Esta distribuição *a priori* tem dois parâmetros τ e ϑ e tem a seguinte forma:

$$P(\log \beta_i \mid \tau, \vartheta) = \frac{1}{\Gamma(\tau)} \left(\frac{\beta_i}{\vartheta} \right)^{\tau} \exp(-\beta_i / \vartheta) . \tag{3.98}$$

A estimativa resultante da variância para o especialista é então dada por:

$$\sigma_i^2 = \frac{1}{d} \frac{\sum_{t=1}^{N} h_i^{(t)} (y^{(t)} - \mu_i^{(t)})^2 + 2/\vartheta}{\sum_{t=1}^{N} h_i^{(t)}} . \tag{3.99}$$

A interpretação destes hiperparâmetros[5] é que $2/\vartheta$ define a soma do erro *a priori* que se pode esperar para um único especialista, e 2τ define o número *a priori* de observações que se pode esperar de um especialista. WATERHOUSE (1998) re-parametrizou estes parâmetros da seguinte forma:

$$\frac{1}{\vartheta} = \frac{\tau}{2} \sigma_{prior}^2 , \tag{3.100}$$

onde σ_{priori}^2 é o nível de ruído que WATERHOUSE (1998) fixou em 0.01. Além disso, ele definiu τ como uma função do número de nós terminais, na forma:

$$2\tau = 0{,}01 \left(\frac{N}{I} \right), \tag{3.101}$$

onde N é o número de exemplos no conjunto de treinamento e I número de nós terminais . Observe que a escolha dependerá do intervalo e da escala dos dados. Em WATERHOUSE

[5] Significa parâmetros definidos em função de outros parâmetros.

(1998), todos os exemplos apresentados foram normalizados de forma a terem média zero e desvio padrão unitário.

Um problema com o uso de condições *a priori* é o efeito que a escolha dos hiperparâmetros pode ter sobre o modelo depois do treinamento. Por exemplo, se o nível de ruído é fixado muito alto, o modelo pode degenerar na modelagem usando somente um especialista. Idealmente, para cada novo conjunto de dados deve-se executar várias simulações usando valores diferentes para os hiperparâmetros e examinando os parâmetros do modelo. Na prática, isto pode ser difícil quando um número grande de parâmetros está presente. WATERHOUSE (1998) argumenta que a escolha dos hiperparâmetros resumidos acima pode ser adequada para limitar a verossimilhança assim como evitar singularidades, e não conduzir a soluções degeneradas. As condições *a priori* tornam-se muito úteis em situações em que um grande número de especialistas está presente, inclusive na presença de hierarquia, nas quais a contribuição de cada especialista torna-se pequena.

3.13.6 Escolhendo o número de especialistas e a profundidade da hierarquia

Um ponto não abordado até aqui na descrição do método ME e HME é a especificação da arquitetura: a configuração da mistura e a profundidade da hierárquica. Até o momento, foram consideradas mistura de especialistas com arquiteturas fixas, com número de especialistas e redes *gating* fixados antecipadamente e constantes durante o treinamento.

O número de especialistas e redes *gating* em cada modelo é decidido balanceando o número de parâmetros na arquitetura e o número de amostras do conjunto de treinamento. Este é um método simples para seleção da arquitetura, proposto por RASMUSSEN (1996) para especificação do número de neurônios escondidos em perceptrons multicamadas.

Em MEs, somente uma mistura de especialistas é usada. Assim, somente uma rede *gating* é considerada. O número de especialistas I pode ser dado por (WATERHOUSE, 1998):

$$I = \frac{N}{(n+1)(K+1)}, \tag{3.102}$$

onde N é o número de exemplos de treinamento, n é a dimensão de $\mathbf{x}$ e K é a dimensão de $\boldsymbol{y}$. Em HMEs, uma mistura hierárquica de especialistas é usada. O número de folhas para cada nó na hierarquia, por exemplo, pode ser fixada em 2, isto é, a arquitetura toma a forma de uma árvore binária. Neste caso os parâmetros livres são a profundidade da árvore, b, que é dada por (WATERHOUSE, 1998):

$$b = \frac{1}{\log 2}(\log(N + 2(n+1)) - \log(n+1) - \log(K+2)). \qquad (3.\ 103)$$

A escolha da arquitetura baseada na equação acima é grosseira, uma vez que não leva em conta a complexidade da tarefa. Em alguns casos, o número de parâmetros no modelo pode ser grande demais, resultando potencialmente em funções excessivamente complexas. Em outros casos, há poucos parâmetros, resultando em sub-ajuste dos dados. De fato, dada uma função arbitrariamente complexa, não há razão para esperar que um aumento no número de amostras da função deveria requerer mais parâmetros para aproximá-la (WATERHOUSE, 1998).

A melhor solução para selecionar o tamanho do modelo são os métodos construtivos e os métodos de poda. Métodos construtivos geram iterativamente modelos com mais especialistas a partir de modelos com menos especialistas, quando aplicados no contexto de mistura. QUINLAN (1985) aplicou os métodos construtivos no contexto de árvore de decisão.

Métodos de poda, por outro lado, usam uma estratégia oposta: uma arquitetura com grande número de especialistas é avaliada para detectar especialistas com contribuição desprezível ou prejudicial à mistura, que então são removidos e a nova arquitetura é retreinada. Este processo pode também ser repetido iterativamente usando o desempenho sobre o conjunto de validação como critério de parada. Computacionalmente, métodos de poda têm a desvantagem de requererem de partida o treinamento de arquiteturas com grande número de especialistas.

Por causa da própria estrutura em árvore da HMEs, é muito atraente derivar um algoritmo construtivo para esta arquitetura (FRISTCH, 1996). A literatura de aprendizado de máquina oferece uma grande variedade de algoritmos construtivos para classificação e árvore de decisão (QUINLAN, 1985, QUINLAN & RIVEST, 1989, BREIMAN *et al.*, 1984).

Infelizmente, estes requerem a avaliação do ganho da divisão de todos os possíveis nós, usando critérios baseados na entropia (na maioria das vezes) ou na verossimilhança para eventualmente realizar a melhor divisão e descartar todas as outras. WATERHOUSE (1998) aborda o problema de seleção de arquitetura via procedimentos que recursivamente adicionam parâmetros ao modelo, da mesma maneira adotada por um algoritmo construtivo em árvore (BREIMAN *et al.*, 1984). A cada passo do algoritmo construtivo, um nó terminal é dividido em dois pares de nós terminais e uma rede *gating*. Todos os nós são divididos e o desempenho de cada um deles é avaliado. Aquele que produzir o maior acréscimo na verossimilhança do modelo é mantido. Em outras palavras, para escolher um nó a ser dividido, deve-se dividir todos os nós terminais, avaliar o benefício de cada divisão e ficar com a melhor. Depois de cada adição à árvore, a árvore inteira é re-treinada usando o algoritmo EM descrito neste capítulo. FRISCH *et al.* (1997) sugeriu uma abordagem alternativa para construir modelos HME em árvore. Ele definiu um critério de avaliação para registrar o desempenho do especialista sobre os dados de treinamento, permitindo assim a detecção do melhor ponto a ser ramificado, fornecendo parâmetros adicionais que podem ajudar a superar os erros realizados por este especialista.

Vendo a HMEs como um modelo probabilístico dos dados observados, foi separada a verossimilhança dependente da entrada $l(\chi;\Theta)$ da verossimilhança dependente da geração dos dados $l_k(\chi;\theta_k)$, usando a probabilidade de seleção do especialista fornecida pela rede *gating*:

$$l(\chi;\Theta) = \sum_{t=1}^{N} \log P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\Theta) = \sum_{t=1}^{N}\sum_{k=1}^{m} g_k \log P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\theta_k) \qquad (3.104)$$

$$l(\chi;\Theta) = \sum_{t=1}^{N}\sum_{k=1}^{m} \log[P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)},\theta_k)]^{g_k} = \sum_{k=1}^{m} l_k(\chi;\theta_k) \qquad (3.105)$$

onde os g_k's são o produto da probabilidade da *gating* ao longo do caminho do nó raiz para o k-ésimo especialista, isto é, g_k é a probabilidade de que o especialista k seja responsável pela geração dos dados observados (observe que os g_k's somam em um). A verossimilhança escalonada, dependente do especialista $l_k(\chi;\Theta)$, pode ser usada como medida para o desempenho de um especialista dentro de sua região de responsabilidade. O especialista com menor l_k é selecionado a partir da árvore e dividido, fornecendo parâmetros

randômicos para a rede *gating*, e uma perturbação randômica dos parâmetros do especialista para gerar um par de novos especialistas. A nova rede é então treinada até que um erro mínimo seja encontrado junto ao conjunto de validação, para o qual uma nova forma de árvore é proposta.

Conforme pode ser observado na descrição acima, todos os algoritmos construtivos envolvem um processo de escolha do nó que será dividido em um outro par de nós. No momento da decisão, o algoritmo construtivo deve escolher o nó que dará o maior acréscimo na verossimilhança. Há dois tipos de métodos diferentes que foram explorados na literatura para decidir como escolher o nó: aqueles baseados na inferência estatística antes que a divisão seja realizada (método estatístico *a priori*), e aqueles baseados na busca iterativa através de todas as possíveis divisões (busca iterativa). Muitos métodos de particionamento recursivo (BREIMAN *et al.*, 1984; FRIEDMAN, 1991) utilizam um método de busca iterativa. Em WATERHOUSE & ROBINSON (1996) foi utilizada uma busca iterativa, enquanto SAITO & NAKANO (1996) e FRITSCH (1997) usaram métodos estatísticos *a priori*. Críticos do método de busca iterativa têm apontado que o custo computacional de avaliação para cada separação pode ser grande se os parâmetros de cada divisão têm que ser treinados a fim de serem avaliados. Entretanto, o maior problema com os métodos estatísticos *a priori* parece ser a falta de garantia da melhor escolha, isto é, aquele que dará melhor acréscimo local no desempenho do modelo. WATERHOUSE (1998) usou o método de busca iterativa e argumenta que este demonstrou empiricamente ser o método mais robusto para escolha do nó a ser dividido.

Pode-se constatar que não existe um consenso sobre o método de escolha do melhor nó a ser dividido. Nesta tese, serão utilizadas arquiteturas fixas definidas por tentativa e erro ou definidas de uma forma aproximada utilizando as equações (3.102) e (3.103). Extensões voltadas para o ajuste da topologia da HMEs fazem parte das perspectivas futuras da tese.

3.14 Considerações Finais

Neste capítulo, foi apresentada uma descrição detalhada das arquiteturas MEs e HMEs, com uma notação unificada e fornecendo uma interpretação probabilística do conjunto de equações desenvolvidas, na tentativa de motivar e mostrar a capacidade de tal ferramenta. No Capítulo 6, os resultados aqui discutidos serão utilizados para sustentar algumas possíveis extensões de mistura de especialistas, tendo SVMs como componentes, seja na forma de especialistas, seja no papel da *gating*.

Capítulo 4

Máquinas de Vetores-Suporte

Resumo: As máquinas de vetores-suporte (SVMs, do inglês *Support Vector Machines*) representam uma generalização do algoritmo *Generalized Portrait* desenvolvido na Rússia nos anos sessenta (VAPNIK & LERNER, 1963; VAPNIK & CHERVONENKIS, 1964). Uma abordagem similar, empregando programação linear ao invés de programação quadrática, foi desenvolvida na mesma época nos Estados Unidos, principalmente por MANGASARIAN (1964; 1969). A estratégia de aprendizado baseada em vetores-suporte é solidamente fundamentada na teoria de aprendizado estatístico, ou teoria VC (Vapnik-Chervonenkis), que vem sendo desenvolvida ao longo das últimas três décadas (VAPNIK & CHERVONENKIS, 1974; VAPNIK, 1982; VAPNIK, 1995) visando a proposição de técnicas de aprendizado de máquina que buscam maximizar a capacidade de generalização. Em sua formulação mais recente, a abordagem SVM foi concebida nos laboratórios da AT&T por Vapnik e co-autores (BOSER *et al.*, 1992; GUYON *et al.*, 1993; CORTES & VAPNIK, 1995; SCHÖLKOPF *et al.*, 1995; VAPNIK *et al.*, 1997), sendo orientada a aplicações práticas. Os experimentos pioneiros envolveram reconhecimento óptico de caracteres (OCR, do inglês *Optical Character Recognition*), com grande sucesso (SCHÖLKOPF *et al.*, 1996; SCHÖLKOPF *et al.*, 1998). BURGES (1998) apresenta um tutorial abrangente acerca da aplicação de SVM junto a problemas de classificação de padrões. Com sua extensão para o tratamento de problemas de regressão, a abordagem SVM mostrou-se altamente competitiva, com destaque para as

aplicações envolvendo predição de séries temporais (WEIGEND *et al*., 1995). Um panorama do estado da arte em aprendizado SVM foi publicado por SCHÖLKOPF (1999).

4.1 Introdução

O problema de modelagem de dados empíricos é pertinente a muitas aplicações de engenharia. Em modelagem de dados empíricos, é usado um processo de indução para construir um modelo matemático capaz de expressar as relações de entrada-saída, a partir do qual são deduzidas respostas ainda não observadas. Basicamente, a quantidade e qualidade dos dados disponíveis governam o desempenho deste modelo empírico. Por corresponder a uma técnica de aprendizado baseada em dados amostrados, a finitude do conjunto de dados fornece tratabilidade computacional, mas conduz a uma amostragem esparsa do espaço de entrada. Conseqüentemente, o problema de aprendizado tende a ser mal condicionado (POGGIO *et al.*, 1985) no sentido de Hadamard (HADAMARD, 1923), ou seja, não há dependência contínua dos dados e o processo de indução de modelos não possui solução única. Quando uma multiplicidade de soluções candidatas são igualmente admissíveis, a capacidade de generalização dos modelos resultantes passa a representar um critério de qualidade capaz de atenuar o efeito do mal condicionamento.

No entanto, modelos matemáticos com capacidade de aproximação universal, como as redes neurais artificiais (HORNIK *et al.*, 1989) ainda não são dotados de algoritmos de treinamento capazes de maximizar a capacidade de generalização de uma forma sistemática, o que pode levar a um sobre-ajuste do modelo aos dados. Por operar no espaço original dos dados, em que as não-linearidades presentes e a complexidade intrínseca do problema não são conhecidas a priori, os algoritmos de otimização para ajuste de parâmetros e as ferramentas estatísticas adotadas para seleção de modelos podem induzir modelos com baixa capacidade de generalização.

Este cenário representa a principal motivação para a proposição de máquinas de vetores-suporte (SVMs). VAPNIK (1995) apresentou uma formulação que engloba o princípio de minimização do risco estrutural (SRM, do inglês *structural risk minimization*),

sendo que VAPNIK *et al.* (1997) mostraram que este princípio é superior ao princípio de minimização do risco empírico (ERM, do inglês *empirical risk minimization*), sendo que este último é empregado no projeto de redes neurais artificiais. O princípio SRM envolve a minimização de um limite superior sobre o erro de generalização, enquanto que o princípio ERM envolve a minimização do erro sobre os dados de treinamento. Logo, modelos de aprendizado de máquina baseados no princípio SRM tendem a apresentar uma maior habilidade para generalizar bem frente a dados não observados, sendo este um dos principais propósitos do aprendizado estatístico.

Embora o princípio SRM tenha sido originalmente desenvolvido para solucionar problemas de classificação, ele foi estendido com sucesso para tratar problemas de regressão (VAPNIK *et al.*, 1997).

Na literatura, a terminologia para SVM é confusa. O termo máquina de vetores-suporte (SVM, do inglês *Support Vector Machine*) é usado tipicamente para descrever classificação por vetores-suporte, enquanto que regressão de vetores-suporte (SVR, do inglês *Support Vector Regression*) é empregado para descrever regressão por vetores-suporte. Nesta tese, o termo SVM será adotado tanto para classificação como para regressão, e os termos classificação por vetores-suporte (SVC, do inglês *Support Vector Classification*) e regressão por vetores-suporte (SVR). Além da questão da terminologia, não há na literatura uma formulação para SVM que adote uma notação unificada, sendo esta uma das principais contribuições deste capítulo.

Nas próximas seções, são apresentados alguns conceitos de teoria estatística, com destaque para risco funcional, minimização do risco empírico e minimização do risco estrutural. Em seguida, é descrita a formulação básica de SVM para classificação e regressão. Para ilustrar as propriedades mais relevantes da metodologia, alguns exemplos de aplicação são analisados em cada caso. Os problemas de seleção de parâmetros e de seleção de características são então abordados, finalizando-se o capítulo com a descrição de algumas formulações alternativas existentes na literatura, as quais também foram consideradas nos experimentos da tese.

4.2 Fundamentação Teórica

A fim de proporcionar um embasamento teórico para o problema de aprendizado em SVM, são introduzidas a seguir algumas definições e suposições a respeito do processo de geração dos dados.

4.2.1 Amostras Independentes e Identicamente Distribuídas

Considera-se que os dados de treinamento, representados pelo par de matrizes de entrada-saída ($\mathbf{X}$, $\mathbf{Y}$), já foram obtidos e são distribuídos independente e identicamente de acordo com uma medida de probabilidade $p(\mathbf{x},y)$. A matriz $\mathbf{X}$ é de dimensão $N \times n$ e a matriz $\mathbf{Y}$ é de dimensão $N \times 1$, com N representando o número de amostras, n a dimensão do espaço de entrada e 1 a dimensão do espaço de saída. A generalização para o caso de múltiplas saídas será realizada na seção 4.10.

Com isso, a amostra $(\mathbf{x}_i, y_i)$ que compõe a i-ésima linha das matrizes ($\mathbf{X}$, $\mathbf{Y}$), com $\mathbf{x}_i \in \Re^n$ e $y_i \in \Re$, é independente das demais amostras e segue $p(\mathbf{x},y)$. Esta suposição é mais restritiva do que pode parecer à primeira vista. Por exemplo, dados de séries temporais tendem a não satisfazer esta condição, uma vez que as observações são tipicamente dependentes e as propriedades estatísticas podem depender do índice i (SMOLA *et al.*, 2000).

4.2.2 Funcional de Risco

VAPNIK (1995) definiu o funcional $R(g)$ como a probabilidade de classificação incorreta, tendo como argumento uma função $g(\cdot)$. Esta definição pode ser generalizada para argumentos na forma de funções de regressão e de decisão limitada. Isto produz o que se chama de funcional de risco.

Seja $c(\mathbf{x},y,g(\mathbf{x})): \Re^n \times \Re \times \Re \rightarrow [0,\infty)$ uma função custo e $p(\mathbf{x},y)$ uma medida de probabilidade como descrita na seção anterior. Então o funcional de risco para funções de regressão $g: \Re^n \rightarrow \Re$ é definido na forma:

$$R(g) = \int_{\Re^n \times \Re} c(\mathbf{x}, y, g(\mathbf{x}))p(\mathbf{x}, y)dxdy\,. \tag{4.1}$$

Como um exemplo usual de função custo junto a problemas de regressão, tem-se:

$$c(\mathbf{x}, y, g(\mathbf{x})) = c(y, g(\mathbf{x})) = (y - g(\mathbf{x}))^2\,. \tag{4.2}$$

Considerando N amostras sujeitas a $p(\mathbf{x},y)$, o funcional de risco empírico é dado por:

$$R_{emp}(g) = \frac{1}{N}\sum_{i=1}^{N} c(\mathbf{x}_i, y_i, g(\mathbf{x}_i))\,. \tag{4.3}$$

Para funções de decisão limitadas, na forma g: $\Re^n \rightarrow \{-1,1\}$, freqüentemente adota-se o erro de classificação 0-1 como função custo:

$$\begin{cases} c(\mathbf{x}, y, g(\mathbf{x})) = 0 & \text{se } g(\mathbf{x}) = y \\ c(\mathbf{x}, y, g(\mathbf{x})) = 1 & \text{se } g(\mathbf{x}) \neq y \end{cases}. \tag{4.4}$$

Neste caso, obtém-se o funcional de risco definido em SMOLA *et al.* (2000), o qual corresponde à probabilidade de classificação incorreta, como segue:

$$R(g) = \Pr\{g(\mathbf{x}) \neq y\}. \tag{4.5}$$

O funcional de risco empírico é dado diretamente pela equação (4.3), que vai corresponder ao erro médio de treinamento.

Finalmente, necessita-se de uma quantidade chamada *margem de erro*, que é dada pela proporção de amostras de treinamento que têm margem menor que ρ. A função custo assume a forma:

$$\begin{cases} c(\mathbf{x}, y, g(\mathbf{x})) = 0 & \text{se } |g(\mathbf{x}) - y| < \rho \\ c(\mathbf{x}, y, g(\mathbf{x})) = 1 & \text{se } |g(\mathbf{x}) - y| \geq \rho \end{cases}, \tag{4.6}$$

e o funcional de risco empírico continua sendo dado pela equação (4.3).

Esta estimativa empírica de risco conta as amostras como um erro se:

- Em regressão: se o erro de aproximação é maior ou igual a ρ;
- Em classificação: se a classificação é correta, mas com margem menor que ρ, ou se a classificação é incorreta.

O objetivo é minimizar o funcional de risco $R(g)$ apresentado na equação (4.1). Entretanto, geralmente $p(\mathbf{x},y)$ é desconhecida. Assim, recorre-se à minimização do

funcional de risco empírico $R_{emp}(g)$ apresentado na equação (4.3), o qual é baseado nos dados de treinamento.

Como o conjunto de dados de treinamento é finito, obter $R_{emp}(g) = 0$ não implica $R(g) = 0$. Considerando uma densidade amostral fixa na definição dos dados de treinamento, a complexidade intrínseca da função g vai determinar o relacionamento entre $R_{emp}(g)$ e $R(g)$, conforme será discutido nas seções seguintes.

4.2.3 Aspectos da Teoria de Aprendizado Estatístico

Esta seção contém uma breve introdução à teoria de aprendizado estatístico. Para uma consideração mais abrangente, veja VAPNIK (1998).

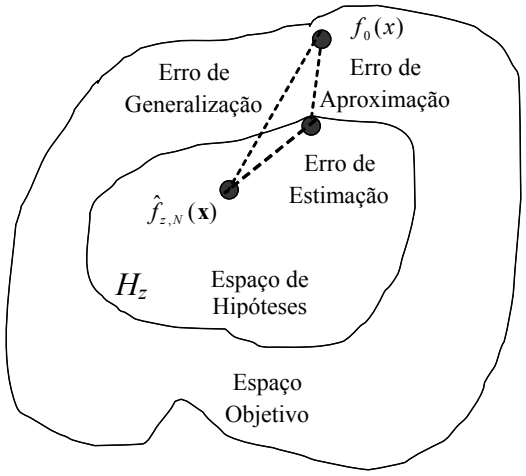


Figura 4.1 – Tipos de erro de modelagem. O parâmetro z está associado à complexidade do modelo (por exemplo, número de neurônios na camada intermediária de uma rede neural artificial) e o parâmetro N indica a quantidade de dados de treinamento.

O objetivo em modelagem funcional é escolher um modelo, a partir do espaço de hipóteses H_z, que é próximo, em relação a alguma medida de erro, à função fundamental $f_0(\cdot)$ no espaço objetivo. Neste processo, dois tipos de erros podem surgir:

- **Erro de Aproximação**: é uma conseqüência do espaço de hipóteses H_z não conter todo o espaço objetivo, de modo que a função fundamental $f_0(\cdot)$ pode residir fora de H_z. Uma

escolha ruim de H_z irá resultar em um erro de aproximação grande, o qual é referido como discordância do modelo.

- **Erro de Estimação**: é o erro devido ao processo de aprendizado, que pode levar à seleção de um modelo que não seja o melhor possível dentro de H_z.

Em conjunto, estes erros formam o erro de generalização. Conforme já mencionado, o que se busca é a minimização do risco empírico, dado na forma:

$$\hat{f}_{z,N} = \arg \min_{f \in H_z} R_{emp}(f) \quad . \tag{4.7}$$

A minimização do risco empírico deve ser consistente no sentido de que:

$$\lim_{N \to \infty} R_{emp}(f) = R(f) . \tag{4.8}$$

VAPNIK (1998) afirma que a equação (4.8) é verdadeira a partir da lei dos grandes números. Entretanto, em virtude da existência do espaço de hipóteses H_z, a condição de consistência deve ser reformulada como segue:

$$\lim_{N \to \infty} \min_{f \in H_z} R_{emp}(f) = \min_{f \in H_z} R(f) , \tag{4.9}$$

que não é válida para qualquer H_z.

4.2.4 Dimensão VC

A dimensão VC é um índice escalar que mede a complexidade intrínseca de uma classe de funções. Na Figura 4.2, é apresentada uma forma de obtenção da dimensão VC para funções lineares no $\Re^2$, ou seja, retas. Conclui-se que a dimensão VC de retas no $\Re^2$ é 3, pois 3 é o número máximo de amostras que podem ser corretamente classificadas por uma reta, para qualquer padrão de rotulação binária que as amostras podem admitir. Para 4 ou mais amostras, existem padrões de rotulação que não possibilitam uma classificação correta por intermédio de uma reta.

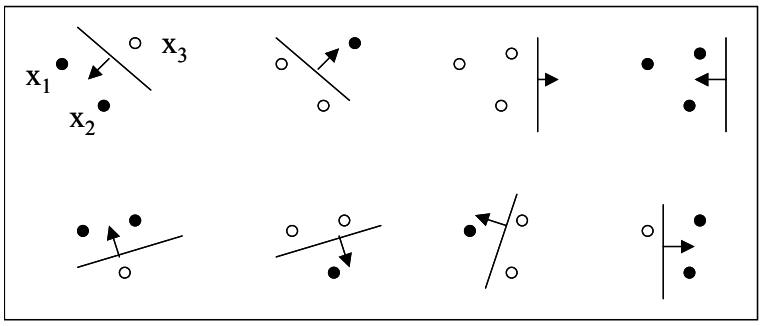


Figura 4.2 – Possibilidades de rotulação de três amostras no $\Re^2$ e a classificação realizada por uma função linear

Definição 4.1 (Dimensão VC - VAPNIK & CHERVONENKIS, 1971). A dimensão VC de uma classe de funções é *h* se e somente se existe um conjunto de amostras $\{\mathbf{x}_t\}_{t=1}^{h}$ tal que, para qualquer uma das 2^h configurações possíveis de rotulação binária, as amostras podem ser corretamente classificadas e não existe nenhum conjunto $\{\mathbf{x}_t\}_{t=1}^{q}$ com $q > h$ satisfazendo esta propriedade.

De forma genérica, para funções lineares no $\Re^n$, com $n \geq 2$, a dimensão VC é n+1. Existem funções com dimensão VC infinita (VAPNIK, 1995).

4.2.5 Minimização do Risco Empírico

No caso de problemas de classificação envolvendo duas classes, também denominados problemas de classificação binária, a tarefa de aprendizagem a partir de amostras de treinamento pode ser formulada como segue:

Dada uma classe de funções de decisão:

$$\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}, \text{ com } f_\lambda : \Re^n \rightarrow \{-1,+1\}$$

onde Λ é um conjunto de parâmetros que faz com que $\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}$ corresponda ao espaço de hipóteses H_z, e um conjunto de exemplos:

$$(\mathbf{x}_1, y_1),...,(\mathbf{x}_N, y_N), \text{ com } \mathbf{x}_i \in \Re^n, y_i \in \{-1,+1\}, i = 1,...,N,$$

obtidos de uma distribuição desconhecida $p(\mathbf{x},y)$, deseja-se encontrar uma função f_λ que forneça o menor valor possível para o funcional de risco:

$$R(\lambda) = \int \left| f_\lambda(\mathbf{x}) - y \right| p(\mathbf{x}, y) dx dy \,. \tag{4.10}$$

O risco esperado é, portanto, uma medida de quão bem uma hipótese candidata prediz o valor correto de y para uma amostra $\mathbf{x}$.

Uma vez que $p(\mathbf{x},y)$ é desconhecida, não há como calcular o funcional de risco $R(\lambda)$, de modo que deve-se recorrer a uma estimativa obtida dos dados de treinamento, ou seja, o funcional de risco empírico, como segue:

$$R_{emp}(\lambda) = \frac{1}{N} \sum_{i=1}^{N} \left| f_\lambda(\mathbf{x}_i) - y_i \right| . \tag{4.11}$$

Uma vez que a lei dos grandes números garante que $R_{emp}(\lambda)$ converge em probabilidade para $R(\lambda)$, o princípio da minimização do risco empírico se sustenta na suposição de consistência de que a minimização de $R_{emp}(\lambda)$ vai fornecer o mínimo de $R(\lambda)$. Como mostrado por VAPNIK & CHERVONENKIS (1971, 1991) e VAPNIK (1995), a consistência é válida se e somente se a convergência em probabilidade de $R_{emp}(\lambda)$ para $R(\lambda)$ é substituída pela convergência uniforme em probabilidade. Eles também mostraram que a condição necessária e suficiente para consistência do princípio de minimização do risco empírico é a limitação da dimensão VC do espaço de hipóteses H_z, ou seja, da classe de funções $\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}$.

A teoria de convergência uniforme em probabilidade também proporciona um limitante para a variação do risco esperado $R(\lambda)$ em função do risco empírico $R_{emp}(\lambda)$. Um limitante típico que acontece com probabilidade $1-\delta$, tem a seguinte forma:

$$R(\lambda) \le R_{emp}(\lambda) + \sqrt{\frac{h \ln\left(\frac{2N}{h} + 1\right) - \ln\left(\frac{\delta}{4}\right)}{N}} \quad \forall \lambda \in \Lambda \,, \tag{4.12}$$

onde h é a dimensão VC de $\{f_\lambda(\mathbf{x}) : \lambda \in \Lambda\}$. Deste limitante, deduz-se que a minimização do risco esperado, o qual vai estar associado ao desempenho de generalização do modelo resultante, requer que tanto o risco empírico como a razão entre a dimensão VC e o número

de amostras de treinamento tem que ser pequena. Uma vez que o risco empírico é usualmente uma função decrescente de h (Vapnik, 1995), para um dado número de amostras de treinamento, há um valor ótimo da dimensão VC. A escolha de um valor apropriado para h, que em muitos casos é controlado através do número de parâmetros livres do modelo, é crucial para se obter bom desempenho, especialmente quando o número de amostras de treinamento é pequeno.

O limitante apresentado na inequação (4.12) será substituído por um outro princípio de indução, conforme motivação apresentada na próxima seção.

4.2.6 Minimização do Risco Estrutural

A técnica de minimização do risco estrutural (SRM, do inglês *Structural Risk Minimization*), desenvolvida por VAPNIK (1982), é uma tentativa de tratamento o problema da escolha de uma dimensão VC apropriada. Da inequação (4.12) fica evidente que um valor pequeno para o risco empírico $R_{emp}(\lambda)$ não necessariamente implica em um valor pequeno para o risco esperado $R(\lambda)$. Um princípio diferente de indução, chamado *princípio de minimização do risco estrutural*, ou simplesmente princípio SRM, é baseado na observação de que ambos os termos do lado direito da inequação (4.12) deveriam ser pequenos. Portanto, tanto a dimensão VC quanto o risco empírico deveriam ser minimizados simultaneamente. A fim de implementar o princípio SRM é necessário propor uma estrutura aninhada para o espaço de hipóteses, na forma:

$$H_1 \subset H_2 \subset \cdots \subset H_k \subset \cdots, \tag{4.13}$$

com a propriedade de que $h(k) \leq h(k+1)$ onde $h(k)$ é a dimensão VC de H_k.

Então, desconsiderando o fator logarítmico na inequação (4.12), resulta o seguinte problema de otimização:

$$\min_{H_k} \left(R_{emp}(\lambda) + \sqrt{\frac{h(k)}{N}} \right). \tag{4.14}$$

Embora extensões sejam admitidas, está sendo considerado aqui que o processo fundamental sendo modelado é determinístico e que existem múltiplas entradas a partir das quais é desejado predizer uma única saída.

O princípio SRM é claramente bem fundamentado matematicamente, mas pode ser difícil de ser implementado pelas seguintes razões:

1. pode ser difícil calcular a dimensão VC de H_k, além do fato de que há somente um número pequeno de classes de funções para as quais é sabido como calcular a dimensão VC;
2. mesmo admitindo a viabilidade de obtenção da dimensão VC de H_k, o problema de minimização da expressão (4.14) pode ser de difícil solução.

A próxima seção vai mostrar que, embora não seja trivial controlar a dimensão VC da técnica de aprendizado durante a fase de treinamento, isto será adequadamente realizado pela abordagem SVM, a qual busca minimizar simultaneamente um limitante para a dimensão VC e um funcional de risco empírico. A formulação da abordagem SVM será apresentada inicialmente para o tratamento de problemas de classificação, mostrando inclusive como sua implementação esta relacionada à programação quadrática. Posteriormente, a formulação será estendida para problemas de regressão.

4.3 Máquinas de Vetores-Suporte para Classificação

O problema de classificação pode ser tomado como um problema de duas classes, sem perda de generalidade (GUNN, 1998). A tarefa é separar duas classes por uma função que é induzida a partir das amostras de treinamento. O objetivo é produzir um classificador que apresenta um bom desempenho junto a amostras não-observadas durante o treinamento, isto é, um classificador que generaliza bem. Considere o exemplo da Figura 4.3. Observe que há vários classificadores lineares possíveis que podem separar as amostras disponíveis sem nenhum erro, mas há somente um que maximiza a margem (maximiza a distância entre o classificador e a amostra mais próxima de cada classe). Este classificador linear é chamado *hiperplano de separação ótimo*, pois, ao menos intuitivamente, espera-se que este

hiperplano generalize melhor que os demais, quando amostras não utilizadas durante a fase de treinamento devem ser classificadas.

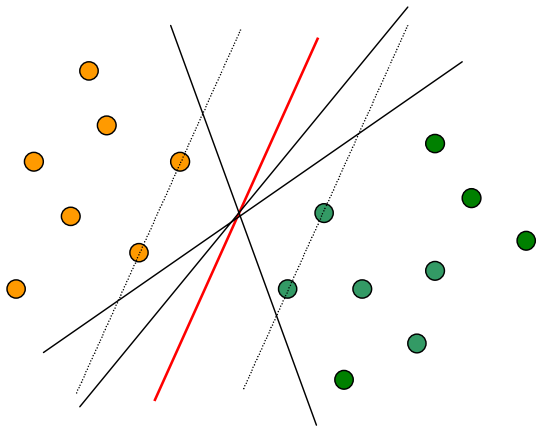


Figura 4.3 – Hiperplano de separação ótim (em vermelho), com os seus hiperplanos-suporte (tracejados).

Analisando a Figura 4.3, pode-se observar que o hiperplano ótimo foi construído tendo como suporte dois outros hiperplanos (linhas tracejadas), os quais por sua vez passam por alguns pontos para ambas as classes. Estes pontos serão chamados de *vetores-suporte*.

4.3.1 O Hiperplano de Separação Ótimo

Considere o problema de separação do conjunto de N vetores de treinamento pertencentes a duas classes linearmente separáveis:

$$(\mathbf{x}_1, y_1),...,(\mathbf{x}_N, y_N), \text{ com } \mathbf{x} \in \Re^n \text{ e } y \in \{-1,+1\}. \tag{4.15}$$

Pata tanto, um hiperplano na forma:

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0, \text{ com } \mathbf{w} \in \Re^n \text{ e } b \in \Re, \tag{4.16}$$

representa a fronteira discriminante, de modo que o lado caracterizado por $(\mathbf{w} \cdot \mathbf{x}) + b > 0$ representa uma classe, e o lado caracterizado por $(\mathbf{w} \cdot \mathbf{x}) + b < 0$ representa a outra classe. A notação $(\mathbf{w} \cdot \mathbf{x})$ indica produto interno entre **w** e **x.**

O conjunto de vetores é dito ser separado otimamente pelo hiperplano se ele é separado sem erro e se a distância do hiperplano à(s) amostra(s) mais próxima(s) de uma classe, somada à distância do hiperplano à(s) amostra(s) mais próxima(s) da outra classe,

denominada margem de separação $\rho(\mathbf{w},b)$, é máxima. Os vetores **x** das amostras de treinamento vão obedecer:

$$\left|(\mathbf{w}\cdot\mathbf{x})+b\right| \geq \frac{\rho(\mathbf{w},b)}{2}, \tag{4.17}$$

pois o hiperplano ótimo deve ficar à igual distância da(s) amostra(s) mais próxima(s) das duas classes.

Na forma em que se encontra a equação (4.17), $\rho(\mathbf{w},b)$ pode ser tão grande quanto se queira, bastando atuar na norma de **w** e no valor de *b*. É necessário, portanto, impor alguma restrição, ou sobre a norma de **w**, ou sobre a própria margem de separação $\rho(\mathbf{w},b)$. Fixando a margem de separação em $\rho(\mathbf{w},b) = 2$, o problema de maximizar a margem se transforma num problema equivalente de minimizar a norma de **w** (VAPNIK, 1999). Resulta então um hiperplano canônico, onde **w** e *b* devem ser tais que:

$$\min_{\mathbf{w}} \left|(\mathbf{w}\cdot\mathbf{x})+b\right| = 1. \tag{4.18}$$

A equação (4.18) força a norma de **w** a ser igual à $2/\rho(\mathbf{w},b)$. Sendo assim, ao minimizar a norma de **w**, obtém-se a maximização da margem de separação $\rho(\mathbf{w},b)$. Logo, como $y \in \{-1,+1\}$, um hiperplano de separação na forma canônica deve satisfazer às seguintes restrições junto às amostras de treinamento em (4.15):

$$\begin{cases} (\mathbf{w}\cdot\mathbf{x}_i)+b \geq +1, & \text{para } y_i = +1,\ i = 1,\ldots,N \\ (\mathbf{w}\cdot\mathbf{x}_i)+b \leq -1, & \text{para } y_i = -1,\ i = 1,\ldots,N \end{cases} \tag{4.19}$$

Estas podem ser combinadas em um conjunto de desigualdades:

$$y_i\left[(\mathbf{w}\cdot\mathbf{x}_i)+b\right] \geq 1, \text{ com } i = 1,...,N. \tag{4.20}$$

A distância[1] $d(\mathbf{w},b;\mathbf{x})$ de um vetor $\mathbf{x} \in \Re^n$ ao hiperplano pode ser expressa na forma:

$$d(\mathbf{w},b;\mathbf{x}) = \frac{\left|(\mathbf{w}\cdot\mathbf{x})+b\right|}{\|\mathbf{w}\|}. \tag{4.21}$$

Com isso, a margem de separação, sujeita às restrições da expressão (4.20), é dada por:

[1] Seja um ponto $P(x_1,y_1)$ e uma reta *r* definida por $ax+by+c=0$. A distância do ponto *P* a reta *r* é definida por: $d(a,b;P) = \dfrac{|ax_1+b_1+c|}{\sqrt{a^2+b^2}}$

$$\rho(\mathbf{w},b) = \min_{\{x_i,y_i=1\}} d(\mathbf{w},b;\mathbf{x}_i) + \min_{\{x_j,y_j=-1\}} d(\mathbf{w},b;\mathbf{x}_j)$$

$$\rho(\mathbf{w},b) = \min_{\{x_i,y_i=1\}} \frac{\left|\mathbf{w}\cdot\mathbf{x}_i + b\right|}{\left\|\mathbf{w}\right\|} + \min_{\{x_j,y_j=-1\}} \frac{\left|\mathbf{w}\cdot\mathbf{x}_j + b\right|}{\left\|\mathbf{w}\right\|}$$

$$\rho(\mathbf{w},b) = \frac{1}{\left\|\mathbf{w}\right\|}\left(\min_{\{\mathbf{x}_i,y_i=1\}} \left|\mathbf{w}\cdot\mathbf{x}_i + b\right| + \min_{\{x_j,y_j=-1\}} \left|\mathbf{w}\cdot\mathbf{x}_j + b\right| \right)$$

$$\rho(\mathbf{w},b) = \frac{2}{\left\|\mathbf{w}\right\|}. \tag{4.22}$$

Assim, o hiperplano que separa otimamente os dados é aquele que minimiza:

$$\Phi(\mathbf{w}) = \frac{1}{2}\left\|\mathbf{w}\right\|^2. \tag{4.23}$$

A independência de *b* se explica, pois mudando *b*, o hiperplano se move na sua direção normal. Conseqüentemente, a margem permanece inalterada, mas o hiperplano não será ótimo, uma vez que estará mais próximo de uma classe que de outra. Minimizar a expressão na equação (4.23) é equivalente a implementar o princípio SRM (GUNN, 1998). Isso será ilustrado a seguir, tomando-se um limite arbitrário *A* para a norma de **w**. Logo:

$$\left\|\mathbf{w}\right\| \le A, \tag{4.24}$$

e, das equações (4.20) e (4.21), resulta:

$$d(\mathbf{w},b;\mathbf{x}) \ge \frac{1}{A}. \tag{4.25}$$

Conseqüentemente, o hiperplano não pode ser mais próximo que $1/A$ para qualquer amostra de treinamento e, intuitivamente, a Figura 4.4 mostra como são reduzidas as possibilidades de hiperplanos candidatos, eliminando aqueles que tendem a generalizar pior.

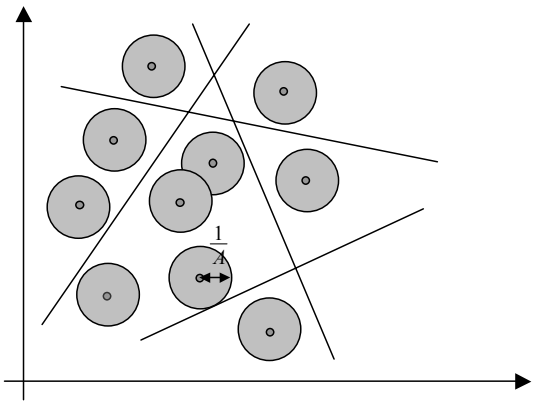


Figura 4.4 – Os candidatos a hiperplano canônico devem manter uma distância mínima de $1/A$ das amostras.

A dimensão VC, aqui denominada pela letra *h*, dos candidatos a hiperplano ótimo assume a forma (VAPNIK, 1995):

$$h \leq \min[R^2 A^2, n] + 1, \tag{4.26}$$

onde *n* é a dimensão do espaço e *R* é o raio de uma hiper-esfera circundando todas as amostras. Assim minimizar a equação (4.23) é equivalente a minimizar o limite superior da dimensão VC. Este é um dos resultados mais expressivos deste capítulo, pois viabiliza a aplicação do princípio SRM.

Tomando o problema de otimização da equação (4.23), juntamente com as restrições da expressão (4.20), resulta um problema quadrático com restrições, cuja solução é o ponto de sela do funcional lagrangeano (MINOUX, 1986):

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i \{[(\mathbf{w} \cdot \mathbf{x}_i) + b] y_i - 1\}, \tag{4.27}$$

onde α_i ($i = 1,...,N$) são os multiplicadores de Lagrange. O lagrangeano da equação (4.27) tem que ser minimizado com respeito a **w** e *b* e maximizado com respeito a $\alpha_i \geq 0$. O problema dual equivalente assume a forma (BAZAARA *et al.*, 1993):

$$\max_{\alpha} W(\alpha) = \max_{\alpha} \{\min_{\mathbf{w},b} L(\mathbf{w}, b, \alpha)\} \tag{4.28}$$

O mínimo do lagrangeano com respeito a **w** e *b* é dado por:

$$\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i y_i = 0 \,, \tag{4.29}$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i \, \mathbf{x}_i y_i \,.$$

Assim, das equações (4.27), (4.28) e (4.29), resulta o seguinte problema dual:

$$\max_{\alpha} W(\alpha) = \max_{\alpha} \left[-\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^{N} \alpha_i \right] \tag{4.30}$$

cuja solução é dada por:

$$\alpha^* = \arg\min_{\alpha} \left[\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^{N} \alpha_i \right] \tag{4.31}$$

com restrições:

$$\begin{cases} \alpha_i \geq 0, i = 1, ..., N \\ \sum_{i=1}^{N} \alpha_i y_i = 0 \end{cases} \tag{4.32}$$

Com a solução do problema de minimização da expressão (4.31), considerando as restrições apresentadas nas expressões (4.32), resulta:

$$\mathbf{w}^* = \sum_{i=1}^{N} \alpha_i^* \mathbf{x}_i y_i \tag{4.33}$$

$$b^* = -\frac{1}{2} \left(\left(\mathbf{w}^* \cdot \mathbf{x}_r \right) + \left(\mathbf{w}^* \cdot \mathbf{x}_s \right) \right), \tag{4.34}$$

onde $\mathbf{x}_r$ e $\mathbf{x}_s$ são os vetores-suporte de cada classe, satisfazendo,

$$\alpha_r, \alpha_s > 0, y_r = 1, y_s = -1 \tag{4.35}$$

Analisando a equação (4.35), pode-se observar que os vetores x_r e x_s, correspondem matematicamente os pontos nos quais os multiplicadores α são maiores que zeros. Logo os vetores-suporte (SVs, do inglês *Support Vectors*) serão aqueles pontos nos quais $\alpha > 0$.

O classificador assume a forma:

$$f(\mathbf{x}) = sign(\mathbf{w}^* \cdot \mathbf{x} + b^*) \,, \tag{4.36}$$

onde $sign(\cdot)$ é a função sinal, definida como segue:

$$sign(y) = \begin{cases} +1, & y \geq 0 \\ -1, & y < 0 \end{cases}. \tag{4.37}$$

Como uma alternativa, um classificador suave que interpole linearmente a margem pode ser adotado:

$$f(x) = g(\mathbf{w}^* \cdot \mathbf{x} + b) \text{ , onde } g(y) = \begin{cases} -1, & y < -1 \\ y, & -1 \le y \le 1 \\ +1, & y > 1 \end{cases}. \tag{4.38}$$

Este pode ser mais apropriado que o classificador da equação (4.36), porque produz saída de valor real entre −1 e +1 quando o classificador é consultado dentro da margem, onde não residem amostras de treinamento para este caso. Das condições de Kuhn-Tucker:

$$\alpha_i^* [y_i(\mathbf{w}^* \cdot \mathbf{x}_i + b^*) - 1] = 0 \tag{4.39}$$

e, assim, somente os pontos $\mathbf{x}_i$ que satisfazem:

$$y_i(\mathbf{w}^* \cdot \mathbf{x}_i + b^*) = 1 \tag{4.40}$$

terão multiplicador de Lagrange diferente de zero, razão pela qual são denominados *vetores-suporte*(SVs). Se as amostras de treinamento são linearmente separáveis, todos os SVs devem estar exatamente à distância unitaria do hiperplano ótimo, de modo que o número de SVs é tipicamente muito pequeno. Conseqüentemente, o hiperplano é determinado por um pequeno subconjunto de amostras de treinamento. As demais amostras não participam da definição do hiperplano ótimo.

Com isso, uma outra interpretação possível para a abordagem SVM é a busca de uma conjunto mínimo de amostras de treinamento capazes de conduzir à indução do classificador. Se as amostras de treinamento são linearmente separáveis, a seguinte igualdade será válida:

$$\left\| \mathbf{w}^* \right\|^2 = \sum_{i=1}^{N} \alpha_i^* = \sum_{SVs} \alpha_i^* = \sum_{SVs} \sum_{SVs} \alpha_i^* \alpha_j^* (\mathbf{x}_i \cdot \mathbf{x}_j) y_i y_j \ , \tag{4.41}$$

Onde $\sum_{SVs}$ implica a inclusão no somatório dos índices correspondentes aos vetores-suporte. Assim, da equação (4.26) a dimensão VC do classificador é limitada por:

$$h \le \min\left[R^2 \sum_{SVs} \alpha_i^*, n \right] + 1, \tag{4.42}$$

e se os dados de treinamento, **x,** forem normalizados para residir no intervalo $[-1,1]^n$, definindo um hipercubo unitário, então:

$$h \leq 1 + n \times \min\left[\sum_{SVs} \alpha_i^*, 1\right] \tag{4.43}$$

4.3.2 Exemplos para dados Linearmente Separáveis

Para ilustrar a aplicação da abordagem SVM junto a problemas de classificação linearmente separáveis, considere o conjunto de treinamento da Tabela 4.1, sendo que a solução é apresentada na Figura 4.5. As linhas tracejadas descrevem os hiperplanos-suporte e os pontos circulados em azul claro representam os vetores-suporte (SVs), todos posicionados nos dois hiperplanos-suporte. As classes são representadas pelos pontos em vermelho e azul, respectivamente. Esta interface gráfica faz parte do pacote de software em Aprendizado de Máquina disponível em http://www.lbic.fee.unicamp.br/machine_learning/package1. Neste ponto, boa parte das informações contidas na Figura 4.5 ainda não foram discutidas e devem ser desconsideradas.

$\mathbf{x}_1$	$\mathbf{x}_2$	Classe
1	1	2
3	3	1
1	3	1
3	1	2
2	2,5	1
3	2,5	2
4	3	2

Tabela 4.1 – Dados de classificação linearmente separáveis.

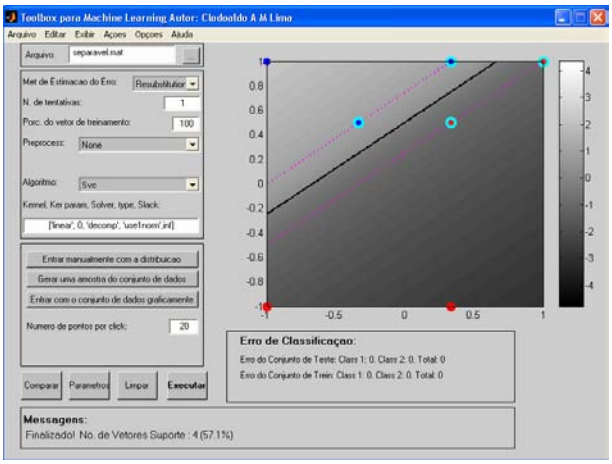


Figura 4.5 – Hiperplano ótimo para classes linearmente separáveis e posicionamento dos vetores-suporte.

4.3.3 O Hiperplano de Separação Ótimo Generalizado

Até este ponto do capítulo, toda a discussão esteve restrita ao caso em que as amostras de treinamento são linearmente separáveis. Entretanto, em geral este não será o cenário a ser encontrado, sendo mais comum a situação ilustrada na Figura 4.6, onde a obtenção de uma margem de separação expressiva vai levar à ocorrência de erros de classificação.

Deve ficar claro, neste ponto do texto, que estamos introduzindo a abordagem SVM para classificação por um hiperplano *no espaço original das amostras de treinamento*. Sem a incorporação do conceito de espaço de características, a ser apresentado na seção 4.4, a abordagem SVM permanece incompleta, sem seus recursos mais poderosos. Somente com base no que já foi apresentado, conclui-se que se trata de uma abordagem para classificação linear apenas. Por hora é necessário manter esta condição para permitir um acompanhamento da linha de argumentação do capítulo.

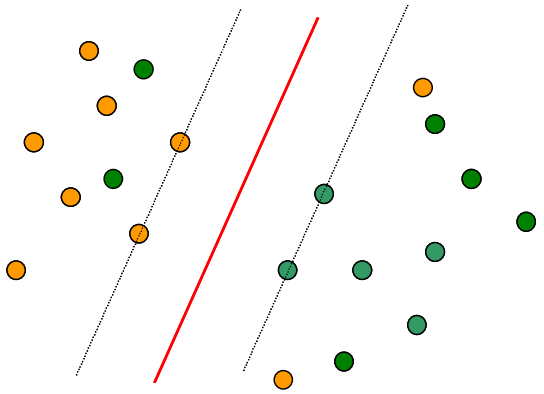


Figura 4.6 – Hiperplano de separação ótimo generalizado, admitindo erros de classificação.

Há duas formas de generalizar a abordagem SVM de modo a admitir erros de classificação, que são dependentes do conhecimento a priori do problema e da estimativa do ruído presente nas amostras de treinamento. No caso em que é esperado (ou possivelmente conhecido) que o hiperplano não pode separar corretamente os dados, um método baseado na introdução de uma função-custo associada com o erro de classificação é desejado. Embora funções mais complexas pudessem ser empregadas para descrever a fronteira, CORTES & VAPNIK (1995) introduziram variáveis não negativas $\xi_i \geq 0$ e uma função de penalidade, na forma:

$$F_\sigma(\xi) = \sum_{i=1}^{N} \xi_i^\sigma, \sigma > 0 \tag{4.44}$$

onde cada ξ_i, i=1,...,N, indica o erro de classificação associada a cada amostra de treinamento. A função-objetivo do problema de otimização é agora composta de um termo associado à dimensão VC do classificador e um outro termo vinculado ao erro de classificação. Sendo assim, o hiperplano de separação ótimo generalizado é determinado pelo vetor **w**, que minimiza o funcional:

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i, \tag{4.45}$$

sujeito às restrições:

$$\xi_i \geq 0$$

$$y_i[(\mathbf{w} \cdot \mathbf{x}_i) + b] \geq 1 - \xi_i, \quad i = 1, \ldots, N, \tag{4. 46}$$

onde C é um valor a ser arbitrado pelo usuário. As restrições em (4.46) representam uma generalização das restrições apresentadas em (4.20).

A solução para o problema de minimização da função-objetico na equação (4.45), sujeito às restrições em (4.46), é dado pelo ponto de sela do lagrangeano (MINOUX, 1986):

$$L(\mathbf{w},b,\alpha)=\frac{1}{2}(\mathbf{w}^T\mathbf{w})+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\alpha_i\{[(\mathbf{x}_i\cdot\mathbf{w})+b]y_i-1+\xi_i\}-\sum_{i=1}^{N}\beta_i\xi_i\ , \qquad (4.47)$$

onde α_i,β_i são multiplicadores de Lagrange. O lagrangeano tem que ser minimizado em relação a $\mathbf{w}$, b e ξ e minimizado em relação a $\alpha_i,\beta_i\geq 0$. O problema primal associado à minimização da expressão (4.47) pode ser transformado em seu problema dual. O problema dual é dado por:

$$\max_{\alpha} W(\alpha)=\max_{\alpha}\{\min_{\mathbf{w},b,\xi} L(\mathbf{w},b,\xi,\alpha,\beta)\} \qquad (4.48)$$

O mínimo em relação a: $\mathbf{w}$, b e ξ do lagrangeano deve atender às seguintes condições:

$$\begin{cases}\dfrac{\partial L}{\partial b}=0\Rightarrow\displaystyle\sum_{i=1}^{N}\alpha_i y_i=0\\[2ex] \dfrac{\partial L}{\partial \mathbf{w}}=0\Rightarrow\mathbf{w}=\displaystyle\sum_{i=1}^{N}\alpha_i\,\mathbf{x}_i y_i\ .\\[2ex] \dfrac{\partial L}{\partial \xi_i}=0\Rightarrow\alpha_i+\beta_i=C\end{cases} \qquad (4.49)$$

As condições de Karush-Khun-Tucker para o problema primal pode ser rescrita como (FLETCHER, 1987):

$$\begin{cases}\alpha_i\{[(\mathbf{x}_i\cdot\mathbf{w})+b]y_i-1\}=0\\ \alpha_i\geq 0\\ \beta_i\xi_i=0\\ \beta_i\geq 0,\ \ i=1,\ldots,N\end{cases}\ . \qquad (4.50)$$

Assim, das equações (4.47), (4.48), (4.49) e (4.50), resulta:

$$\max_{\alpha} W(\alpha)=\max_{\alpha}\left[-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i\cdot\mathbf{x}_j)+\sum_{i=1}^{N}\alpha_i\right] \qquad (4.51)$$

A solução é tal que:

$$\alpha^* = \arg\min_{\alpha}\left[\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\left(\mathbf{x}_i \cdot \mathbf{x}_j\right) - \sum_{k=1}^{N}\alpha_k\right] \tag{4.52}$$

sujeito a:

$$\begin{cases} 0 \le \alpha_i \le C, i = 1,..., N \\ \sum_{i=1}^{N}\alpha_i y_i = 0 \end{cases}. \tag{4.53}$$

A solução para este problema de minimização é idêntica ao caso separável, exceto pela modificação do limite de excursão dos multiplicadores de Lagrange.

Uma limitação da abordagem de CORTES & VAPNIK (1995) é que não foi apresentado um procedimento sistemático para determinação do coeficiente de penalidade *C*. Em algumas circunstâncias, *C* pode ser diretamente relacionado com um parâmetro de regularização (GIROSI, 1997; SMOLA & SCHÖLKOPF, 1998). BLANZ *et al.* (1996) usa um valor de $C = 5$, mas é razoável argumentar que o valor de *C* deve variar com o nível de ruído presente nas amostras de treinamento. Métodos para proposição de valores adequados para *C*, assim como técnicas que permitem deixar a abordagem menos sensível, em termos de desempenho, à escolha adequada de *C*, serão apresentadas na seção 4.20 e no Capítulo 5, respectivamente.

4.3.4 Exemplo para Dados Não-Linearmente Separáveis

Duas amostras adicionais são incorporadas aos dados da Tabela 4.1, deixando o problema de classificação não-linearmente separável. O novo conjunto de amostras é apresentado na Tabela 4.2.

x_1	x_2	Classe
1	1	0
3	3	1
1	3	1
3	1	0
2	2,5	1
3	2,5	0
4	3	0
1.5	1,5	1
1	2	0

Tabela 4.2 – Dados de classificação não-linearmente separáveis.

Adotando C=10, o resultado da aplicação da abordagem SVM é mostrado na Figura 4.7. Neste caso, a orientação do hiperplano e a largura da margem são diferentes daquelas apresentadas na Figura 4.5. Os vetores-suporte (SVs) não necessariamente vão residir sobre os hiperplanos-suporte. Os SVs são os pontos circulados em azul claro, enquanto os pontos circulados por um quadrado representam aqueles que contribuem para o erro de classificação.

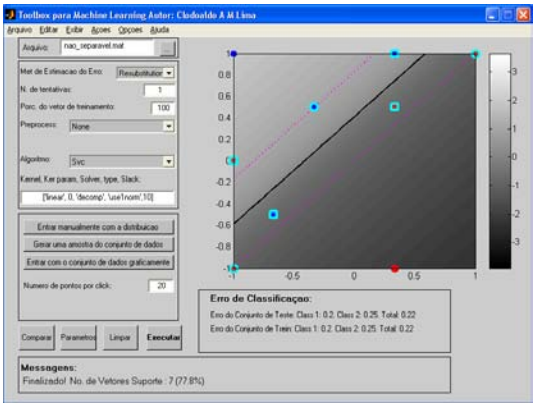


Figura 4.7 – Exemplo de hiperplano de separação ótimo generalizado, para amostras de treinamento não-linearmente separáveis e com C = 10.

Analisando a Figura 4.8, observa-se que, quando $C \rightarrow \infty$, a solução converge para a solução obtida pelo hiperplano de separação ótimo (sobre estes dados não separáveis) (veja Figura 4.5).

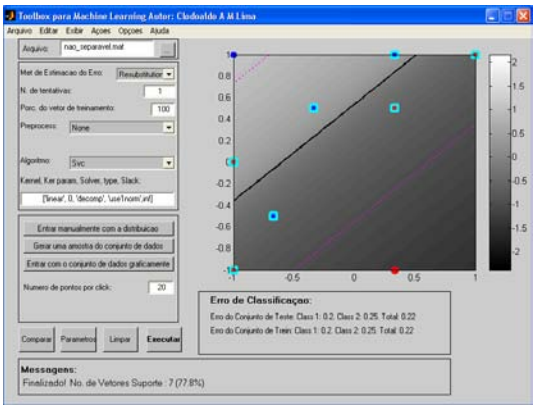


Figura 4.8 – Exemplo de hiperplano de separação ótimo generalizado, para amostras de treinamento não-linearmente separáveis e com $C \rightarrow \infty$.

Quando C se aproxima de zero, a solução passa a ser dominada pelo termo de maximização da margem, Figura 4.9. Quase todos os multiplicadores de Lagrange assumirão o valor de C. Há agora menos ênfase sobre a minimização do erro de classificação, produzindo uma margem que aumenta com o decréscimo de C.

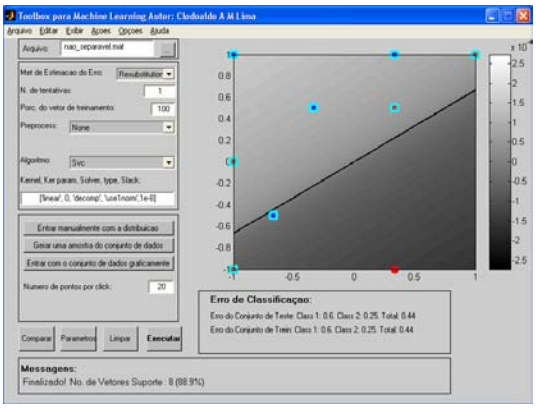


Figura 4.9 – Exemplo de hiperplano de separação ótimo generalizado, para amostras de treinamento não-linearmente separáveis e com $C = 10^{-8}$.

4.4 Espaço de Características de Alta Dimensão

No caso em que uma fronteira linear é inapropriada, o SVM pode mapear o vetor de entrada **x**, em um espaço de características de alta dimensão é não linear. Após o

mapeamento o SVM constrói um hiperplano de separação ótimo neste espaço de alta dimensão, Figura 4.10. Construído este hiperplano no espaço de alta dimensão, os dados são então mapeados para o espaço de saída.

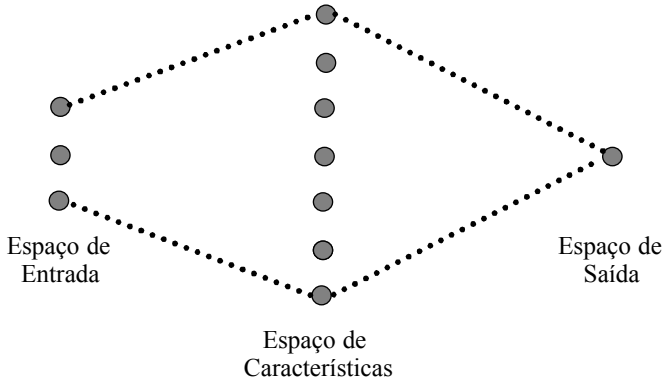


Figura 4.10 – Mapeamento do espaço de entrada em um espaço de características de alta dimensão e do espaço de características para o espaço de saída.

Há algumas restrições sobre o mapeamento não-linear que pode ser empregado (veja seções 4.5.1), mas verifica-se que a maioria das funções comumente empregadas em regressão não-linear é válida. Entre os mapeamentos válidos estão aqueles sintetizados a partir de polinômios, funções de base radial e algumas funções sigmoidais (GUNN, 1998). O problema de otimização da equação (4.52) torna-se:

$$\alpha^* = \arg\min_{\alpha}\left[\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K\left(\mathbf{x}_i,\mathbf{x}_j\right)-\sum_{k=1}^{N}\alpha_k\right], \tag{4.54}$$

onde $K\left(\mathbf{x}_i,\mathbf{x}_j\right)$ é uma função *kernel* (veja mais detalhes na seção 4.5) que representa o mapeamento não-linear que leva ao espaço de características, e as restrições (4.53) não são alteradas, produzindo:

$$\begin{cases}\alpha_i \geq 0, i = 1,...,N \\ \sum_{i=1}^{N}\alpha_i y_i = 0\end{cases}. \tag{4.55}$$

Solucionando o problema de otimização (4.54) com as restrições em (4.55), determina-se os multiplicadores de Lagrange α_i, e os classificadores que implementam o hiperplano de separação ótimo no espaço de característica são dados na forma:

$$f(\mathbf{x}) = sign\left(\sum_{SVs} \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{x}) + b^*\right), \tag{4.56}$$

onde:

$$(\mathbf{w}^* \cdot \mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}), \tag{4.57}$$

$$b^* = -\frac{1}{2} \sum_{SVs} \alpha_i^* y_i \left[K(\mathbf{x}_r, \mathbf{x}_i) + K(\mathbf{x}_s, \mathbf{x}_i)\right]. \tag{4.58}$$

O bias b é calculado aqui usando dois vetores-suporte $\mathbf{x}_r$ e $\mathbf{x}_s$, mas pode ser calculado usando todos os vetores-suporte sobre a margem por questão de estabilidade (VAPNIK *et al.*, 1997). Se o *kernel* contém um termo de bias, o bias do classificador, b, pode ser ajustado dentro da função *kernel*, e assim o classificador assume uma forma mais simples:

$$f(\mathbf{x}) = sign\left(\sum_{SVs} \alpha_i^* y_i K(\mathbf{x}_i, \mathbf{x})\right). \tag{4.59}$$

Muitos dos *kernels* empregados têm um termo de bias e qualquer *kernel* pode ser modelado de modo a apresentar este termo (GIROSI, 1997). Dada uma função *kernel* que contém um termo de bias, o termo b da equação do hiperplano pode ser abandonado. Isto simplifica o problema de otimização, pois pode-se remover a restrição de igualdade em (4.55). As próximas seções discutem em mais detalhes a escolha da função *kernel* e as condições necessárias que devem ser satisfeitas por uma função *kernel* válida.

4.4.1 Exemplo empregando um kernel polinomial

Considere um *kernel* polinomial da forma:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \left((\mathbf{x}_i \cdot \mathbf{x}_j) + 1\right)^2. \tag{4.60}$$

Este kernel mapeia um vetor de entrada bi-dimensional em um espaço de características de dimensão igual a 6, cujas coordenadas são $\left(1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2\right)$, onde x_1, x_2 representam as duas coordenadas do espaço de entrada original **x.** Aplicando o SVM não-linear para dados de treinamento não-linearmente separáveis da Tabela 4.2, produz-se a classificação ilustrada na Figura 4.11 (adotando $C = \infty$). A margem não tem largura constante devido à projeção não-linear no espaço de entrada. A solução é diferente daquela apresentada na Figura 4.8. Aqui os dados de treinamento são agora classificados corretamente. Entretanto, embora o SVM implemente o princípio SRM para qualquer escolha de *kernel*, de modo a generalizar bem, uma escolha cuidadosa da função *kernel* é necessária para produzir uma fronteira de classificação que é topologicamente apropriada. É sempre possível mapear o espaço de entrada em uma dimensão maior que o número de amostras de treinamento, por exemplo, e assim produzir um classificador com nenhum erro de classificação sobre o conjunto de dados de treinamento. Entretanto, este classificador irá generalizar mal. A escolha do *kernel* necessita de investigações adicionais. No Capítulo 5, uma abordagem utilizando ensemble será apresentada com o intuito de tratar estas questões em aberto.

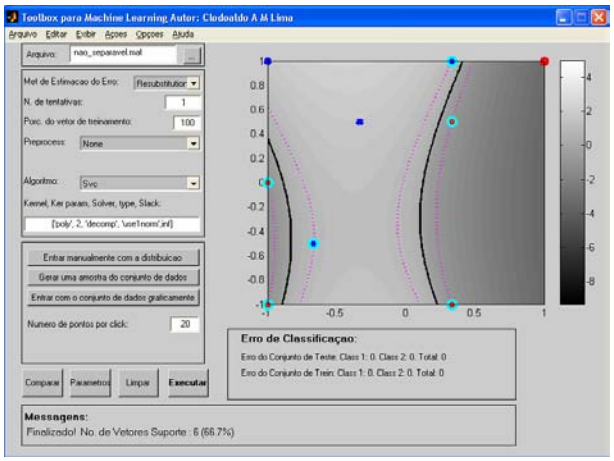


Figura 4. 11 – Mapeamento inverso do hiperplano de separação ótimo, do espaço de características (de dimensão *b*) para o espaço original, usando kernel polinomial.

4.4.2 A questão do parâmetro de penalidade

Tipicamente, os dados serão separáveis linearmente somente em alguns espaços de características de alta dimensão. Pode não fazer sentido tentar separar os dados sem erro de classificação, particularmente quando somente uma quantidade reduzida e ruidosa de dados de treinamento está disponível. Assim, na prática, são empregadas abordagens não-lineares que adotam um limite superior para os multiplicadores de Lagrange.

Fica ainda a questão de como determinar o parâmetro de penalidade *C*, o qual representa um problema similar àquele associado ao coeficiente de penalidade em problemas de regularização. SMOLA & SCHÖLKOPF (1998) mostraram que o parâmetro *C* pode ser diretamente relacionado a um parâmetro de regularização para alguns *kernels*. Um processo de validação cruzada pode ser adotado para determinar este parâmetro, embora outras propostas estejam sendo investigadas na literatura (veja seção 4.20).

Observe que, removendo os padrões de treinamento que não sejam vetores-suporte, a solução não mudará e assim um método rápido para validação pode ser implementado se os vetores-suporte forem esparsos. Nas próximas seções, serão apresentadas algumas abordagens práticas para seleção do parâmetro *C*. Já no Capítulo 5, será utilizada uma abordagem baseada em ensemble para diminuir a dependência do desempenho do classificador em relação à escolha do parâmetro *C*.

4.5 O Papel das Funções *Kernel*

Esta seção discute o método que pode ser usado para construir um mapeamento no espaço de características de alta dimensão usando espaço de reprodução de *kernel* de Hilbert (do inglês *Reproducing Kernel Hilbert Spaces* - RKHS). A idéia da função *kernel* é habilitar as operações a serem executadas no espaço de entrada, evitando assim realizá-las no espaço de características, de maior dimensão. Assim, o produto interno não necessita ser avaliado no espaço de características, amenizando o efeito da maldição da dimensionalidade. Entretanto, o esforço computacional é ainda dependente do número de dados de treinamento e, para proporcionar uma boa distribuição dos dados em problemas de

alta dimensionalidade, um conjunto de treinamento de elevada cardinalidade será necessario.

4.5.1 Funções *Kernel*

Os resultados a seguir são baseados no espaço de reprodução de *kernel* de Hilbert (RKHS) (ARONSZAN, 1950; GIROSI, 1997; HECKMAN, 1997; WAHBA, 1990). Um produto interno no espaço de características tem um *kernel* equivalente no espaço de entrada, na forma:

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)), \tag{4.61}$$

desde que algumas condições sejam satisfeitas. Se K é uma função definida positiva e simétrica que satisfaz as condições de Mercer:

$$\begin{cases} K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{m=1}^{\infty} \alpha_m \varphi_m(\mathbf{x}_i) \varphi_m(\mathbf{x}_j) & \alpha_m \geq 0 \\ \iint K(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_i) g(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_i > 0 & g \in L_2 \,. \\ \int g^2(\mathbf{x}_i) d\mathbf{x}_i < \infty \end{cases} \tag{4.62}$$

onde L_2 é o espaço de norma euclidiana.

Então o *kernel* representa o produto interno no espaço de características. Funções válidas que satisfazem as condições de Mercer são apresentadas abaixo, sendo válidas para todo $\mathbf{x}_i$ e $\mathbf{x}_j$, a menos que se mencione o contrário.

4.5.1.1 Polinômios

Para construir uma regra de decisão polinomial de grau d, é possivel usar o seguinte modelo de geração do *kernel*:

$$\begin{cases} K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} \cdot \mathbf{x}_i)^p \text{ ou} \\ K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x} \cdot \mathbf{x}_i + 1)^p \end{cases} \tag{4.63}$$

O segundo *kernel* é preferível, pois este evita o problema de nulidade da hessiana (matriz utilizada em técnicas de otimização envolvendo a função kernel). Esta função é simétrica e,

re-escrita em coordenadas espaciais, descreve o produto interno no espaço de características que contém todos os produtos $x_{i1},\ldots,x_{ip}$ até o grau p. Usando esta geração de *kernel*, constrói-se uma função de decisão da forma:

$$f(\mathbf{x},\alpha) = sign\left(\sum_{SVs} y_i\alpha_i\left((\mathbf{x}\cdot\mathbf{x}_i)+1\right)^p + b\right), \tag{4.64}$$

que é uma decomposição de polinômios p-dimensionais no espaço de entrada n-dimensional.

Apesar do espaço de características de alta dimensionalidade (polinômios de grau d no espaço de entrada tem $O(n^p)$ parâmetros livres), a estimativa da dimensão VC do subconjunto de polinômios que solucionam problemas práticos (com base em um dado conjunto de treinamento) pode ser baixa (**Teorema 10.3**, VAPNIK, 1995). Se a esperança desta estimativa é baixa, então a esperança da probabilidade de erro é pequena (**Teorema 10.58**, VAPNIK, 1995).

4.5.1.2 Polinômios completos

$$K(\mathbf{x},\mathbf{x}_i) = \left(\frac{(\mathbf{x}\cdot\mathbf{x}_i)}{a} + b\right)^p, \tag{4.65}$$

onde a, b e p são definidos pelo usuário. Este kernel em uma generalização proposta por VAPNIK (1995) para o *kernel* polinomial (veja seção 4.5.1).

4.5.1.3 Funções de base radial gaussiana

Funções de base radial (RBF) usam o seguinte conjunto de regras de decisão:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{N} a_i K(\mathbf{x},\mathbf{x}_i) + b\right), \tag{4.66}$$

onde $K(\mathbf{x},\mathbf{x}_i)$ depende da distância $\|\mathbf{x}-\mathbf{x}_i\|$ entre dois vetores. Para aspectos teóricos envolvendo funções RBF, veja POWELL (1992). A função kernel RBF é uma função

monotônica definida positiva para qualquer σ fixado, e tende a zero quando $\|\mathbf{x}-\mathbf{x}_i\|$ vai para infinito. A função mais popular deste tipo é

$$K(\mathbf{x},\mathbf{x}_i)=\exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|^2}{2\sigma^2}\right), \tag{4.67}$$

Para construir a regra de decisão a partir da definição (4.66), tem-se que estimar:

- O número *N* de centros,
- Os vetores $\mathbf{x}_i$, descrevendo os centros,
- O valor dos parâmetros $\boldsymbol{a_i}$ e b,
- O valor do parâmetro σ.

Os três primeiros passos (determinação do parâmetros σ, *N* e os vetores (centros) $\mathbf{x}_i$, $i=1,\ldots,N$) são geralmente baseados em heurísticas e somente o quinto passo (depois de encontrar estes parâmetros) é determinado minimizando o funcional de risco empírico.

Sabe-se que funções de base radial do tipo (4.66) satisfazem as condições do teorema de Mercer. Portanto, é possível escolher a função $K(\mathbf{x},\mathbf{x}_i)$ com um *kernel* gerando um produto interno em algum espaço de características. Usando este *kernel*, contrói-se uma SVM de funções de base radial. Ao contrário dos métodos clássicos de RBF, com a abordagem SVM todos os quatros tipos de parâmetros são escolhidos automaticamente:

- O número *N* será determinando pelo número de vetores-suporte;
- Os centros serão descritos pelos vetores suporte $\mathbf{x}_i$, $i=1,\ldots,N$;
- Os coeficientes da expansão são dados por $a_i=\alpha_i y_i$ e o valor de *b* pode ser obtido através da equação (4.58);
- O parâmetro de disperção σ da função *kernel* pode ser escolhido minimizando o funcional (4.95) definido na seção 4.8.1.

Esta função local é atrativa no sentido de que os vetores-suporte contribuem com uma função gaussiana local. Considerações adicionais sobre a seleção da dispersão da função de base radial global, σ, usando o princípio SRM, são apresentadas em VAPNIK (1995).

4.5.1.4 Funções de base radial exponencial (ERBF)

Uma função de base radial na forma:

$$K(\mathbf{x},\mathbf{x}_i) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}_i\|}{2\sigma^2}\right) \tag{4.68}$$

onde $\|z\|$, corresponde ao norma de z. O kernel ERBF produz uma solução linear por partes que pode ser atrativa quando se deseja aproximar funções com pouca suavidade.

4.5.1.5 Perceptrons multicamadas

É possível adotar funções *kernel* sigmoidais, na forma:

$$K(\mathbf{x},\mathbf{x}_i) = \frac{1}{1+\exp\{v(\mathbf{x}\cdot\mathbf{x}_i)-c\}} \tag{4.69}$$

Como as funções sigmoidais são adotadas como funções de ativação de redes neurais artificiais, a máquina de vetores-suporte resultante vai corresponder a uma rede neural com uma camada intermediária. Ao contrário dos *kernels* polinomiais ou funções de base radial, que sempre satisfazem a condição de Mercer, o *kernel* sigmoidal satisfaz a condição de Mercer somente para alguns valores dos parâmetros *c* e *v*. Por exemplo se $|\mathbf{x}|=1$, $|\mathbf{x}_i|=1$ os parâmetros *c* e *v* da função sigmoidal têm que satisfazer a desigualdade $c \geq v$. Obedecendo a estas restrições, é possível construir SVMs implementando a seguinte regra:

$$f(\mathbf{x},\alpha) = sign\{\sum_{i=1}^{N}\alpha_i K(\mathbf{x},\mathbf{x}_i)+b\}\,. \tag{4.70}$$

Usando a técnica descrita acima, os seguintes parâmetros são encontrados automaticamente:

- A arquitetura da rede neural de uma camada intermediária, determinando o número *N* de unidades nesta camada intermediária (isto é, o número de vetores-suporte);
- O vetor de pesos dos neurônios da primeira camada (vetores-suporte);
- O vetor de pesos para a camada de saída (valores de α).

4.5.1.6 Série de Fourier

Um importante papel em processamento de sinais é atribuído à expansão em série de Fourier. Nesta seção, constroem-se *kernels* para expansão de Fourier em espaços multidimensionais. Iniciamente será considerado o caso unidimensional.

Suponha que se deseja analisar um sinal unidimensional em termos da sua expansão em série de Fourier. Pode-se mapear as variáveis de entrada **x** em um vetor (2*l*–1)-dimensional.

$$u = (1/\sqrt{2}, sen\mathbf{x}, \ldots, senl\mathbf{x}, \cos\mathbf{x}, \ldots, \cos l\mathbf{x}) . \tag{4.71}$$

Então, para qualquer **x** a expansão de Fourier pode ser considerada como um produto interno neste espaço de características (2*l*–1)-dimensional:

$$f(\mathbf{x}) = (a \cdot u) = \frac{a_0}{\sqrt{2}} + \sum_{k=1}^{l} (a_k senk\mathbf{x} + b_k \cos k\mathbf{x}) . \tag{4.72}$$

Portanto, o produto interno de dois vetores neste espaço tem a forma:

$$K_l(\mathbf{x}, \mathbf{x}_i) = \frac{1}{2} + \sum_{k=1}^{l} (senk\mathbf{x} senk\mathbf{x}_i + \cos k\mathbf{x} \cos k\mathbf{x}_i) . \tag{4.73}$$

Levando em conta a função Dirichet, obtém-se:

$$K(\mathbf{x}, \mathbf{x}_i) = \frac{1}{2} + \sum_{k=1}^{l} \cos k(\mathbf{x} - \mathbf{x}_i) = \frac{\sin(\frac{2l+1}{2})(\mathbf{x} - \mathbf{x}_i)}{\sin(\frac{\mathbf{x} - \mathbf{x}_i}{2})} . \tag{4.74}$$

O *kernel* é definido no intervalo $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$,

Para definir o sinal em termos da expansão de Fourier, a abordagem SVM usa a representação:

$$f(\mathbf{x}, \alpha) = \sum_{i=1}^{N} \alpha_i K_l(\mathbf{x}, \mathbf{x}_i) \tag{4.75}$$

Para construir a SVM para o espaço vetorial *n*-dimensional $\mathbf{x}_i = (x_{i1}, \ldots, x_{in})$, recorre-se um produto de *kernels* unidimensionais, denominado produto tensorial:

$$K(\mathbf{x}, \mathbf{x}_i) = \prod_{k=1}^{n} K(\mathbf{x}_k, \mathbf{x}_{ik}) \tag{4.76}$$

Entretanto, este *kernel* pode conduzir a problemas de regularização, evidenciados pela sua transformada de Fourier (SMOLA & SCHÖLKOPF, 1998).

4.5.1.7 Fourier Regularizado (modo de regularização fraca)

Para o caso unidimensional:

$$K(\mathbf{x},\mathbf{x}_i)=\frac{\pi}{2}\frac{\cosh\left(\pi-\left|\mathbf{x}-\mathbf{x}_i\right|/\gamma\right)}{\sinh\left(\pi/\gamma\right)}\,, \tag{4.77}$$

onde $0\le\left|\mathbf{x}-\mathbf{x}_i\right|<2\pi$ e γ é definido pelo usuário (VAPNIK *et al.*, 1997). Para o caso multidimensional, adota-se a equação (4.76).

4.5.1.8 Fourier Regularizado (modo de regularização forte)

Para o caso unidimensional:

$$K(\mathbf{x},\mathbf{x}_i)=\frac{1-\gamma^2}{2(1-2\gamma\cos(\mathbf{x}-\mathbf{x}_i)+\gamma^2)}\,, \tag{4.78}$$

onde $0\le\left|\mathbf{x}-\mathbf{x}_i\right|<2\pi$ e γ é definido pelo usuário (VAPNIK *et al.*, 1997). Para o caso multidimensional, adota-se a equação (4.76).

4.5.1.9 Splines Lineares

Na aplicação de SVM com splines, o número de junções de splines a ser utilizado não desempenha um papel importante. Portanto para simplificar o cálculo, usa-se splines com um número infinito de junções, definido sobre o intervalo (0,a), $0<c<\infty$, como a expansão:

$$f(\mathbf{x})=\sum_{i=0}^{p}a_i\mathbf{x}^i+\int_0^c s(t)(\mathbf{x}-t)_+^d\,dt\,, \tag{4.79}$$

onde $a_i,\ i=0,\ldots,p$, são valores a determinar e $a(t)$ é uma função desconhecida que define a expansão. Pode-se considerá-la como um produto interno. Portanto, é possível construir o seguinte *kernel* para geração de splines de ordem p com um número infinito de junções:

$$K(\mathbf{x},\mathbf{x}_i)=\int_0^c(\mathbf{x}-t)_+^p(\mathbf{x}-t)_+^p\,dt+\sum_{r=0}^{p}\mathbf{x}_r\mathbf{x}_{ir}\,, \tag{4.80}$$

$$K(\mathbf{x},\mathbf{x}_i)=\int_0^{\mathbf{x}\wedge x'}(\mathbf{x}-t)_+^p(\mathbf{x}-t)_+^p\,dt+\sum_{r=0}^{p}\mathbf{x}_r\mathbf{x}_{ir}\,, \tag{4.81}$$

$$K(\mathbf{x},\mathbf{x}_i)=\int_0^c u^p(u+|\,\mathbf{x}_i-\mathbf{x}\,|)^p\,du+\sum_{r=0}^{p}\mathbf{x}_r\mathbf{x}_{ir}\,, \tag{4.82}$$

$$K(\mathbf{x},\mathbf{x}_i)=\sum_{r=0}^{p}\frac{C_p^r}{2p-r+1}(\mathbf{x}_i\wedge\mathbf{x})^{2p-r+1}\,|\,\mathbf{x}_i-\mathbf{x}\,|^r+\sum_{r=0}^{p}\mathbf{x}_r\mathbf{x}_{ir}\,, \tag{4.83}$$

onde denota-se $\min(\mathbf{x},\mathbf{x}_i)=(\mathbf{x}\wedge\mathbf{x}_i)$. Em particular para spline linear (p = 1) tem-se:

$$K(\mathbf{x},\mathbf{x}_i)=1+\mathbf{x}\mathbf{x}_i+\frac{1}{2}\left|\mathbf{x}-\mathbf{x}_i\right|(\mathbf{x}\wedge\mathbf{x}_i)+\frac{(\mathbf{x}\wedge\mathbf{x}_i)^3}{3} \tag{4.84}$$

Este *kernel* é definido no intervalo [0,1).

Novamente, o *kernel* para splines *n*-dimensionais com um número infinito de junções é o produto de *n kernels* para splines unidimensionais, conforme equação (4.76). Baseado neste *kernel*, pode-se construir uma aproximação por splines (usando a técnica descrita na seção 4.4) que tem a forma:

$$f(\mathbf{x},\alpha)=\sum_{i=1}^{N}\alpha_i K(\mathbf{x},\mathbf{x}_i)\,. \tag{4.85}$$

4.5.1.10 B_n-Splines

B_n-splines representam uma outra formulação spline muito popular. O *kernel* é definido no intervalo [-1,1], e tem a forma:

$$K(\mathbf{x}_i,\mathbf{x}_j)=B_{2n+1}(\mathbf{x}_i-\mathbf{x}_j) \tag{4.86}$$

No entanto, a título de implementação, é necessário definir um equacionamento explícito para B_n-splines, havendo para tanto duas formas: por procedimento iterativo ou uma combinação linear de splines regulares (VAPNIK, 1995).

Para obter tal equacionamento, considera-se a seguinte função B_0-splines (*B*-splines de ordem 0):

$$B_0(u) = \begin{cases} 1, & se \quad |u| \le 0.5 \\ 0, & se \quad |u| > 0.5 \end{cases} \tag{4.87}$$

O B_p-splines de ordem p define-se como uma convolução de duas funções: B_{p-1}-splines e B_0-splines:

$$B_d(u) = \int_{-\infty}^{\infty} B_{d-1}(u-t)B_0(t)dt\ . \tag{4.88}$$

O $B_p(u)$-splines tem a seguinte construção:

$$B_d(u) = \sum_{r=0}^{p+1} \frac{(-1)^r}{r!} C_{d+1}^r \left(u + \frac{p+1}{2} - r \right)_+^p . \tag{4.89}$$

É possível mostrar que ambas as definições descrevem o mesmo objeto (Vapnik, 1995). Usando B_p-splines, pode-se propor o seguinte mapeamento:

$$f(\mathbf{x}, \alpha) = \sum_{i=1}^{N} \alpha_i B_p(\mathbf{x} - t_i)\,, \tag{4.90}$$

onde $t_i,\ i = 1,\ldots,N$, definem as junções da expansão. Uma vez que esta expansão tem a forma de um produto interno, o *kernel* que gera a expansão B_p-splines é dado por:

$$K(\mathbf{x}, \mathbf{x}_i) = \sum_{k=1}^{N} B(\mathbf{x} - t_k)B(\mathbf{x}_i - t_k)\ . \tag{4.91}$$

4.5.1.11 *Kernels* Aditivos

Kernels mais sofisticados podem ser obtidos pela formação de somatórios de *kernels*, uma vez que a soma de duas funções definidas positivas é definida positiva, produzindo:

$$K(\mathbf{x}, \mathbf{x}_i) = \sum_k K_k(\mathbf{x}, \mathbf{x}_i)\ . \tag{4.92}$$

4.6 Efeito de bias implícito e explícito

Já foi mencionado na seção 4.4 que os *kernels* podem ou não conter um bias implícito. A inclusão de um bias na função *kernel* pode conduzir a um método de implementação mais eficiente. Entretanto, as soluções com bias implícito e explícito não

são idênticas. Esta diferença ajuda a realçar as dificuldades com a interpretação de generalização em espaço de características de alta dimensionalidade. A Figura 4.12 compara um *kernel* linear com bias explícito com um polinômio de grau 1 com bias implícito. É evidente que as soluções são diferentes, embora ambas as soluções oferecem uma boa capacidade de generalização.

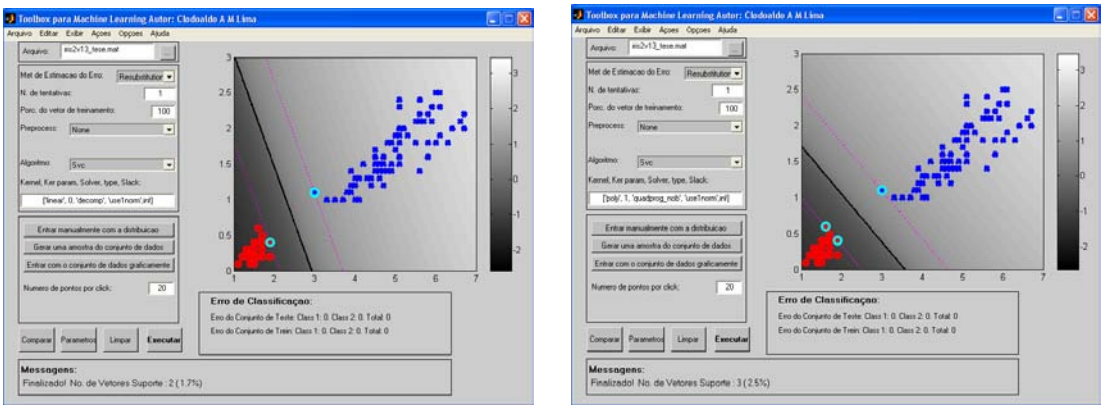


Figura 4.12 – Comparação entre soluções obtidas com bias explícito e implícito.

4.7 Normalização dos dados

Se os dados não forem normalizados, as variáveis de entrada e saída podem excursionar livremente. Com exceção do emprego de métricas que levam em conta a variância das variáveis envolvidas, como a distância de Mahalanobis, todas as demais métricas estariam então sujeitas a resultados polarizados, ou seja, que podem variar com uma variação de escala das variáveis envolvidas (todas ou um subconjunto).

Além da questão da polarização da métrica, a normalização dos dados é necessária para alguns *kernels* em particular, devido ao seu domínio restrito. Observações empíricas também sugerem que a normalização melhora o número de condição da matriz hessiana (matriz de *kernel*) associada ao problema de otimização.

Na Figura 4.13, realiza-se o mesmo experimento da seção anterior, exceto pelo fato de que os dados foram normalizados tanto para o *kernel* linear com bias explícito quanto para o *kernel* polinomial com bias implícito. Comparando os resultados apresentados nas

Figuras 4.12 e 4.13, pode-se concluir que a solução obtida com a normalização dos dados é independente do bias ser implícito ou explícito.

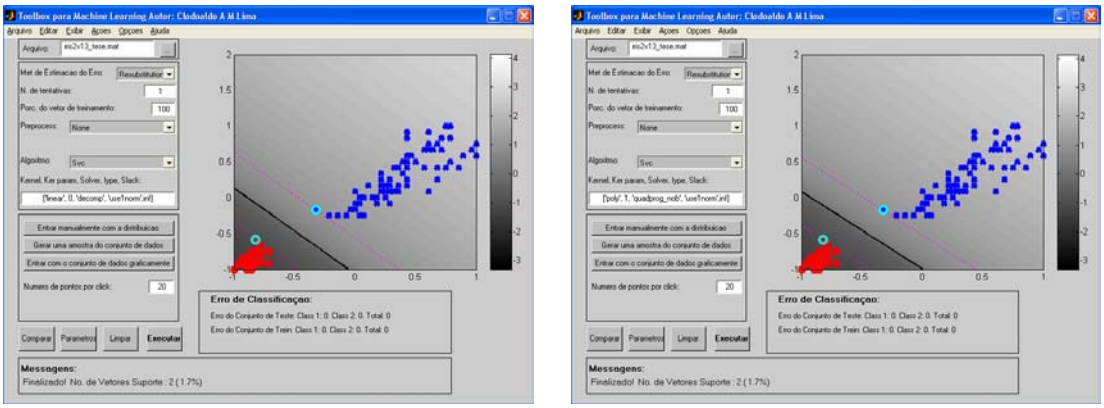


Figura 4.13 – Comparação entre bias explícito e implícito para *kernel* linear, com dados normalizados no intervalo [-1,1].

4.8 Seleção da Função *Kernel*

Dado que há muitos mapeamentos diferentes para um espaço de características, cada um associado a uma função *kernel* que atende ao teorema de Mercer, a questão que surge é: qual é o melhor mapeamento a ser escolhido para um problema em particular?

Esta não é uma questão nova, mas com a inclusão de muitos mapeamentos dentro de uma estrutura é fácil fazer uma comparação. O limite superior sobre a dimensão VC, equação (4.26), é uma alternativa potencial para fornecer meios de comparação entre *kernels*. Entretanto, este limite requer a estimação do raio da hiper-esfera encapsulando os dados em um espaço de características não-linear.

Mesmo que um método teórico com repercussões em aplicações práticas seja proposto para seleção de *kernels*, a menos que este possa ser validado usando um conjunto de teste independente, métodos como *bootstrapping* e validação cruzada, estes permanecerão como alternativas competitivas para seleção de kernel. Nos Capítulos 5 e 6, mostrar-se-á que a utilização de ensemble e de mistura de especialistas, respectivamente, aponta para uma alternativa à escolha do melhor *kernel*.

4.8.1 Seleção de *kernel* utilizando limite superior para a dimensão VC

O limitante superior para a dimensão VC, obtido da equação (4.26), fornece uma estimativa da habilidade de generalização e permite que se escolha um modelo específico (*kernel* específico) de SVM a partir de um conjunto de modelos. Seja $\varphi(\mathbf{x}) = (\varphi_1(\mathbf{x}),\ldots,\varphi_p(\mathbf{x}))$ um vetor no espaço de características e seja $\mathbf{w} = (w_1(\mathbf{x}),\ldots,w_p(\mathbf{x}))$ um vetor de pesos determinando um hiperplano neste espaço, onde p é a dimensão do espaço de característica que pode ser finita ou não.

Considere uma estrutura sobre um conjunto de hiperplanos com um conjunto de funções S_k satisfazendo a condição:

$$\left\lfloor R^2 \|\mathbf{w}\|^2 \right\rfloor \leq k, \tag{4.93}$$

onde R é o raio da menor esfera que contém os vetores $\varphi(\mathbf{x})$, $\|\mathbf{w}\|$ é a norma dos pesos (utiliza-se hiperplanos canônicos no espaço de características com respeito aos vetores $\varphi(\mathbf{x}_i)$, onde $\mathbf{x}_i$ pertence ao conjunto de dados de treinamento) e $\lfloor z \rfloor$ representa o maior valor inteiro de z. De acordo com o **Teorema** 10.3 de VAPNIK (1995) (agora aplicado ao espaço de características), k fornece uma estimativa da dimensão VC do conjunto de funções S_k definido sobre os dados de treinamento.

Considere que a abordagem SVM separa os dados de treinamento

$$y_i[(\varphi(\mathbf{x}_i)\cdot\mathbf{w})+b] \geq 1, \qquad y_i = \{+1,-1\}, \quad i = 1,\ldots,N, \tag{4.94}$$

sem erros e tem uma norma mínima $\|\mathbf{w}\|$. Em outras palavras, a abordagem SVM separa os dados de treinamento usando funções dos elementos S_k com menor estimativa da dimensão VC.

Portanto, pode-se usar o seguinte critério para escolha do melhor modelo:

$$D(R_k, \mathbf{w}_k) = R_k^2 \|\mathbf{w}_k\|^2, \tag{4.95}$$

onde os valores de $\|\mathbf{w}_k\|$ e R_k podem ser calculados a partir dos vetores de treinamento. Repare que, no espaço de características, a igualdade:

$$\frac{1}{\rho_k^2} = \|\mathbf{w}_k\|^2 = \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \left(\varphi(\mathbf{x}_i)\cdot\varphi(\mathbf{x}_j)\right) = \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.96}$$

é verdadeira.

Para definir o raio da menor esfera R_k que inclui os vetores de treinamento, é necessário solucionar o seguinte problema de otimização:

Minimize o funcional $R_k * R_k$

sujeito às restrições

$$\left\|\varphi(\mathbf{x}_i) - c\right\| \leq R_k^2 \quad i = 1, \ldots, N \tag{4.97}$$

onde $\mathbf{x}_i,\ i = 1, \ldots, N$, são os vetores-suporte, e c é o centro da esfera.

Usando a técnica de multiplicadores de Lagrange, é possível obter:

$$R_k^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j \left(\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)\right) \tag{4.98}$$

onde os coeficientes $\beta_i,\ i = 1, \ldots, N$, são uma solução para o seguinte problema de otimização quadrática:

Minimize o funcional

$$W(\beta) = \sum_{i=1}^{N} \beta_i \left(\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)\right) - \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j \left(\varphi(\mathbf{x}_i) \cdot \varphi(\mathbf{x}_j)\right) \tag{4.99}$$

sujeito às restrições

$$\sum_{i=1}^{N} \beta_i = 1, \qquad \beta_i \geq 0 \tag{4.100}$$

Usando a representação kernel, é possível re-escrever o raio da menor esfera na forma:

$$R_k^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{4.101}$$

e o funcional (4.96) fica como segue:

$$W(\beta) = \sum_{i=1}^{N} \beta_i K(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i=1}^{N}\sum_{j=1}^{N} \beta_i \beta_j K(\mathbf{x}_i, \mathbf{x}_j)\,. \tag{4.102}$$

Portanto, para controlar a habilidade de generalização da máquina (minimizar a esperança do erro de teste), deve-se construir o hiperplano de separação que minimiza o funcional:

$$D(R_k, \mathbf{w}_k) = R_k^2 \left\|\mathbf{w}_k\right\|^2 = \frac{R_k^2}{\rho_k^2} \tag{4.103}$$

onde $\|w_k\|^2$ e R_k^2 são definidos pelas equações (4.96) e (4.101), respectivamente.

Escolhendo, entre os diferentes modelos (diferentes *kernels*), o modelo que minimiza a estimativa (4.95), chega-se à melhor proposta para a abordagem SVM.

Assim, a seleção de modelos é baseada na seguinte idéia: entre as diferentes propostas de SVM que separam os dados de treinamento, aquela com menor dimensão VC é a mais indicada a fim de estimar a dimensão VC, pode-se recorrer a limitante superior definida na equação (4.95). Visando obter uma avaliação mais precisa da dimensão VC, VAPNIK (1995) introduziu um método adicional de medida da dimensão VC e mostrou que, independente do método, a menor estimativa da dimensão VC não está necessariamente nessociada à menor dimensionalidade do espaço de características.

4.9 Exemplo de Classificação – Dados Iris

O conjunto de dados IRIS é um conjunto de dados bem conhecido e usado para medir desempenho de algoritmos de classificação. Ele contém 4 atributos (comprimento e largura da pétale e comprimento e largura da sépale) e 3 espécies de gênero *Iris*, e o objetivo é classificar corretamente as amostras de cada uma das 3 espécies baseado nestes 4 atributos. Para visualizar a distribuição das amostras, é possível utilizar somente as duas características que contêm a maior parte da informação a respeito da classe, a saber: comprimento da pétala e largura da pétala. São 50 amostras de cada uma das 3 classes, no entanto, como serão utilizados somente dois atributos, o número de amostras por classe será menor que 50, uma vez que as coordenadas de algumas amostras coincidem. A distribuição dos dados é apresentada na Figura 4.14.

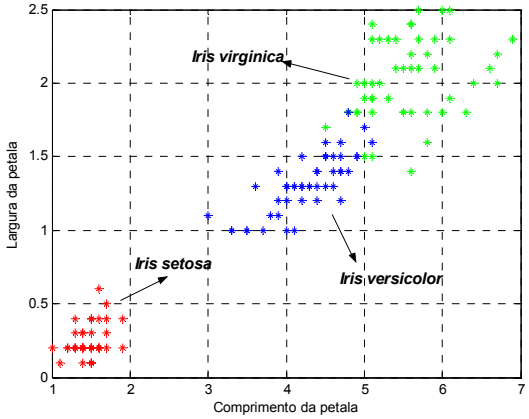


Figura 4.14 – Distribuição do conjunto de dados para amostras do gênero *Iris*, considerando apenas 2 atributos. As espécies são: *Iris virginica*, *Iris versicolor* e *Iris setosa*.

STITSON (1996) foi o primeiro a considerar máquinas de vetores-suporte no tratamento dos dados IRIS, visando avaliar a capacidade de generalização de diferentes configurações. Experimentos similares serão apresentados na seqüência, evidenciando que apenas dois dos 4 atributos estão sendo considerados. Logo, o propósito não é maximizar o poder de acerto do classificador, mas permitir visualizar o resultado graficamente.

As amostras das espécies *Iris setosa* e *Iris versicolor* são separáveis entre si com fronteiras lineares e a solução SVM usando produto interno como função *kernel* é ilustrada na Figura 4.15. Aqui, os vetores-suporte são circulados em azul claro, sendo apenas em número de dois.

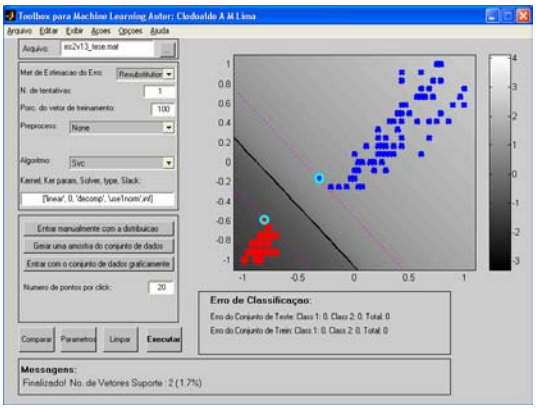


Figura 4.15 – Processo de separação das amostras de *Iris setosa* e *Iris versicolor* usando SVM com *kernel* linear.

Já a separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies não é trivial. De fato, existem amostras de espécies diferentes com valores idênticos para o comprimento e a largura da pétala. A Figura 4.16 ilustra a solução obtida para SVM quando empregando um kernel polinomial de grau 2. As amostras em vermelho estão associadas com a espécie *Iris virginica*.

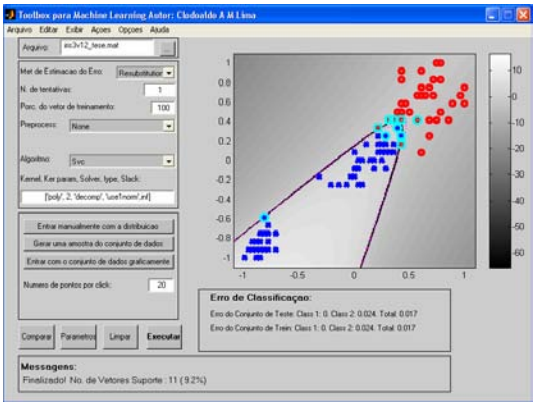


Figura 4.16 – Separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies, usando SVM com *kernel* polinomial (grau 2).

A Figura 4.17 ilustra o uso de polinômio de alto grau (grau 10) para separar as amostras da espécie *Iris virginica* das amostras das outras duas espécies, com nenhum controle da capacidade de modelagem. Com $C \to \infty$, o algoritmo irá ponderar apenas o erro

de treinamento. O espaço de características atinge a dimensão 55. Há evidências de sobre-ajuste devido à natureza da alta dimensão da função *kernel*, que é enfatizada por regiões disjuntas no topo da Figura 4.17.

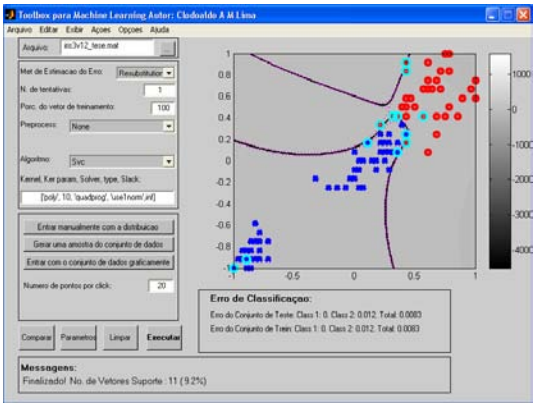


Figura 4.17 – Separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies, usando SVM com *kernel* polinomial (grau 10).

A Figura 4.18 ilustra uma SVM com uma função gaussiana de base radial usando uma variância pré-especificada ($\sigma = 1{,}0$). O resultado é similar àquele obtido com polinômio de grau 2.

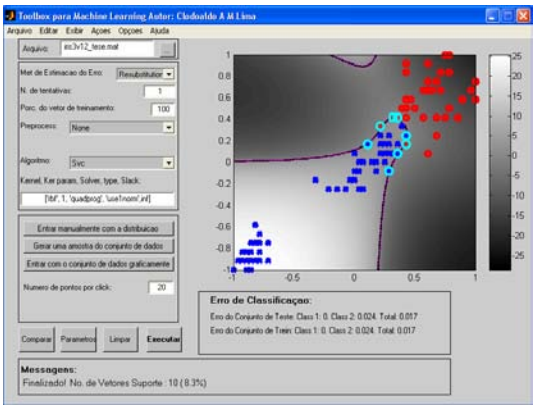


Figura 4.18 – Separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies, usando SVM com funções de base radial (σ=1,0).

A Figura 4.19 ilustra a solução obtida usando um polinômio de grau 2 com alguma tolerância para erros de classificação incorreta ($C = 10$). Esta configuração de parâmetros pode produzir uma solução com bom poder de generalização, enfatizando assim a

importância de se admitir erros de classificação quando se busca uma margem de separação maior.

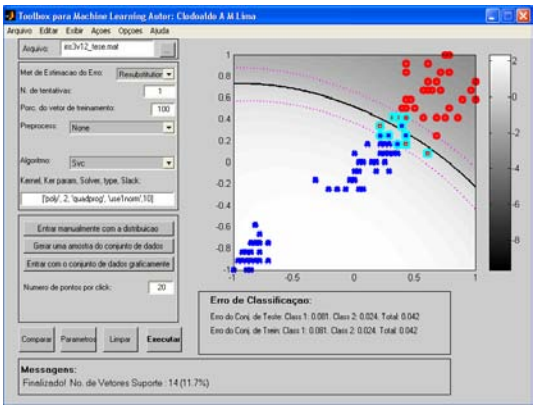
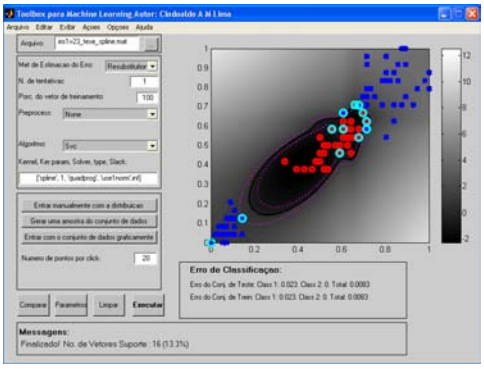
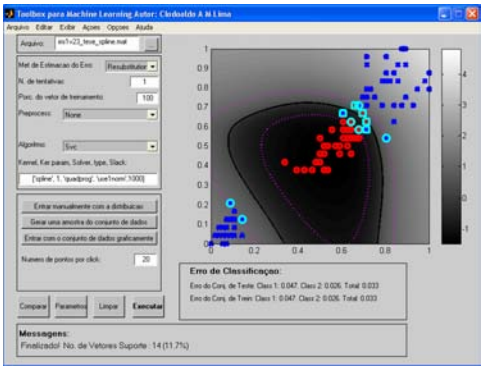


Figura 4.19 – Separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies, usando SVM com *kernel* polinomial (grau 2, $C = 10$).

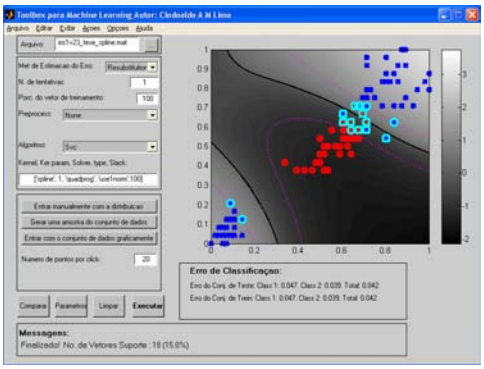
Para visualizar o efeito da tolerância a erros de classificação sobre a topologia da fronteira do classificador e sobre o número de vetores-suporte, a Figura 4.20 mostra os resultado de uma SVM com Spline linear para vários graus de tolerância de classificação incorreta, ou seja, para valores distintos de C. Vale salientar que não é objetivo deste experimento apontar o melhor valor de C, embora a Figura 4.21 apresente o desempenho das máquinas de vetores-suporte em função de C. O que se busca na Figura 4.20 é apontar a influência sobre o resultado final de diferentes valores de C, influência esta que pode conduzir a soluções com desempenho variado em termos de generalização.



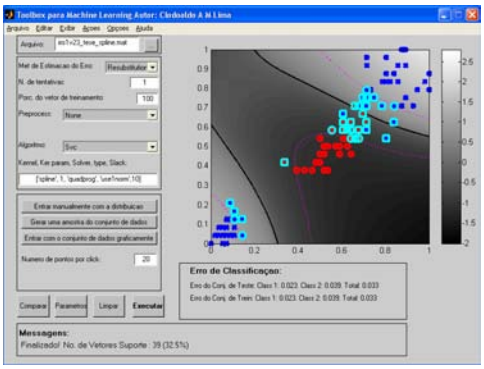
(a) $C = \infty$



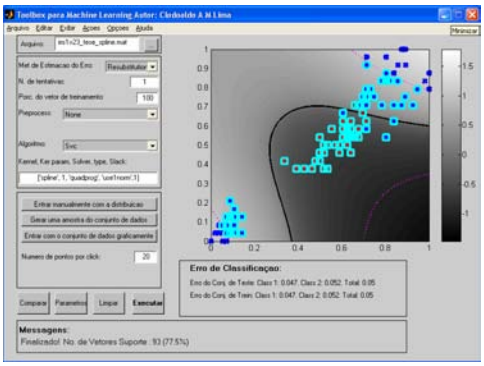
(b) $C = 1000$



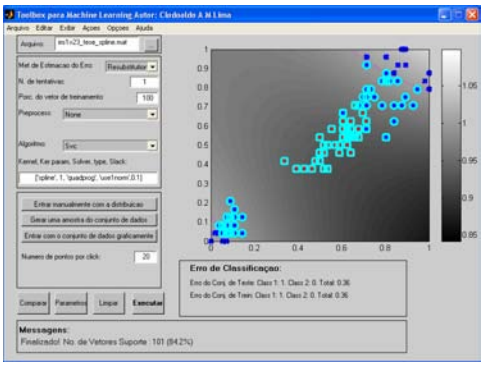
(c) $C = 100$



(d) $C = 10$



(e) $C = 1$



(f) $C = 0{,}1$

Figura 4. 20 – O efeito de *C* sobre a separação das amostras da espécie *Iris virginica* das amostras das outras duas espécies, usando SVM com Spline linear.

A Figura 4.21 ilustra o comportamento do erro de classificação para o kernel spline linear com a variação do parâmetro *C*. O parâmetro *C* foi escolhido uniformemente no intervalo $[10^{-3};10^{+15}]$ e para cada parâmetro escolhido foi utilizado como critério de avaliação do erro a validação cruzada com 10 partições. Validação cruzada com *d* partições significa que o conjunto de treinamento foi dividido em *d* partições e foram realizadas *d* simulações, sendo que para cada simulação $d-1$ partições foram utilizadas no conjunto de treinamento e 1 partição representou o conjunto de teste. Analisando a Figura 4.21, pode-se observar que, a partir de um determinado valor do parâmetro *C*, não ocorre variação

significativa no erro de classificação. No entanto, o intervalo de valores do parâmetro *C* que leva a bons resultados não é conhecido a priori.

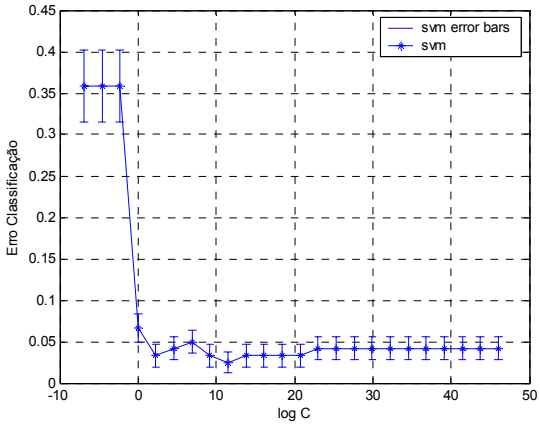


Figura 4. 21 – Resultado para spline linear usando validação cruzada, com variação do parâmetro *C* no intervalo $[10^{-3};10^{+15}]$.

4.10 Metodologias para classificação com múltiplas classes

Conforme já mencionando, a abordagem SVM foi originalmente proposta para classificação binária (CORTES & VAPNIK, 1995). Nos anos que se seguiram, houve iniciativas buscando a extensão da metodologia para classificação com múltiplas classes. Atualmente, há duas propostas bem sucedidas no emprego de SVM com múltiplas classes. Uma abordagem defende construir e combinar vários classificadores binários, enquanto a outra abordagem opta por considerar todos os dados diretamente na formulação de otimização.

A formulação para solucionar problemas de SVM com múltiplas classes em um único passo tem um crescimento no número de variáveis proporcional ao número de classes. Portanto, para métodos SVM com múltiplas classes, ou vários classificadores têm que ser construídos ou um problema de otimização de elevadas dimensões precisa ser resolvido. Assim, em geral é mais custoso computacionalmente solucionar um problema com várias classes que vários problemas de classificação binária.

Na próxima seção, serão apresentados os métodos *um contra todos*, *um contra um* e DAGSVM, que são baseados na solução de vários problemas binários. Em seguida, apresenta-se uma breve introdução aos métodos propostos em VAPNIK (1998), WESTON & WATKINS (1999) e CRAMMER & SINGER (2000), que consideram todas as classes em um único problema de otimização. De acordo com os experimentos numéricos realizados por HSU & LIN (2002), os métodos "um contra um" e DAGSVM são os mais convenientes para uso prático que os outros métodos. Para problemas que envolvem um grande número de atributos, resultados experimentais (HSU & LIN, 2002) mostram que os métodos propostos em VAPNIK (1998) e WESTON & WATKINS (1999) tendem a produzir poucos vetores-suporte, por considerarem todas as variáveis de uma única vez.

4.10.1 Método Um contra Todos

A primeira implementação usando SVM para classificação com múltiplas classes foi provavelmente o método *Um contra Todos (One-Vs-All)*, (BOTTOU, 1994). Ele é um método simples e efetivo (DUBCHAK *et al.*, 1999; BROWN *et al.*, 2000) para problemas com múltiplas classes. Este constrói *K* modelos SVM, onde *K* é o número de classes. O *k*-ésimo SVM é treinado com todos os exemplos da *k*-ésima classe com rótulos positivos, e todos os outros exemplos com rótulo negativo. Assim, considerando *N* amostras de treinamento $(\mathbf{x}_1, y_1),\ldots(\mathbf{x}_N, y_N)$, onde $\mathbf{x}_i \in \Re^n$, $i = 1,\ldots,N$ e $y_i \in \{1,\ldots,K\}$ é a classe de $\mathbf{x}_i$, o *k*-ésimo SVM soluciona o seguinte problema:

$$\min_{w^k, b^k, \xi^k} \quad \frac{1}{2}(\mathbf{w}^k)^T \mathbf{w}^k + C\sum_{i=1}^{N} \xi_i^k \tag{4.104}$$

sujeito a:

$$\begin{cases} (\mathbf{w}^k)^T \varphi(\mathbf{x}_i) + b^k \geq 1 - \xi_i^k & \text{se } y_i = k, \\ (\mathbf{w}^k)^T \varphi(\mathbf{x}_i) + b^k \leq -1 + \xi_i^k & \text{se } y_i \neq k, \\ \xi_i^k \geq 0, \ i = 1,\ldots,N \end{cases} \tag{4.105}$$

onde os dados de treinamento $\mathbf{x}_i$ são mapeados para um espaço de alta dimensão pela função φ, e *C* é um parâmetro de penalidade.

Da mesma forma que na classificação binária, minimizar $\frac{1}{2}(\mathbf{w}^k)^T\mathbf{w}^k$ significa maximizar a margem de separação. Quando os dados não são linearmente separáveis, há um termo de penalidade $C\sum_{i=1}^{N}\xi_i^k$ que visa reduzir o número de erros de treinamento. O conceito básico da abordagem SVM, tanto para classificação binária quanto para classificação com múltiplas classes, é encontrar um equilíbrio entre o termo de regularização $\frac{1}{2}(\mathbf{w}^k)^T\mathbf{w}^k$ e o termo de erro de treinamento.

Após solucionar o problema definido pelas expressões (4.104), há *K* funções de decisão:

$$\begin{matrix}(\mathbf{w}^1)^T\varphi(\mathbf{x})+b^1\\ \vdots\\ (\mathbf{w}^K)^T\varphi(\mathbf{x})+b^K\end{matrix}$$

Diz-se que **x** pertence à classe que tem o valor mais alto da função de decisão:

$$\textit{classe de } \mathbf{x} \equiv \arg\max_{k=1,\ldots,K}\left((\mathbf{w}^k)^T\varphi(\mathbf{x})+b^k\right) \tag{4.106}$$

Na prática, soluciona-se o problema dual daquele definido pela expressão (4.104), cujo número de variáveis é igual ao número de amostras de treinamento na expressão (4.104). Assim *K* problemas quadráticos de *N* variáveis são solucionados.

4.10.2 Método Um contra Todos Único

No processo de classificação usando o método Um contra Todos, o sistema testa um novo exemplo **x** sobre cada um dos *K* classificadores binários para determinar se este pertence a uma dada classe ou não. Este conduz a *K* valores referentes aos *K* classificadores. Idealmente, somente um dos *K* classificadores deveria produzir um resultado positivo, com todos os outros classificadores apontando resultado negativo. Na prática, contudo, muitos classificadores produzirão valores positivos para alguns valores de **x**, indicando que este pode pertencer a mais que uma classe. Trata-se de um resultado de classificação ambígua, o chamado problema do *falso positivo*. Uma das razões para o problema do falso positivo é que a fronteira de decisão entre a classe verdadeira e as outras classes complementares combinadas não podem ser definida claramente.

DING & DUBCHAK (2001) propuseram um novo procedimento para melhorar o método Um contra Todos. A idéia é obter uma predição não ambígua para todos os exemplos de **x**. Isto é realizado reduzindo-se ou eliminando-se os falsos positivos. Eles adicionaram um segundo passo ao método Um contra Todos, aplicando uma classificação discriminante sobre os pares de classificadores entre todas as classes com predição positiva.

Suponha que, para um exemplo **x**, o método Um contra Todos aponte q classificações positivas, isto é, **x** pode pertencer a q classes, com $q \leq K$. Se o número de classes é K, então um passo adicional envolve o treinamento de classificadores para todos os $K(K-1)/2$ pares de classes. Os $q(q-1)/2$ classificadores que envolvem as q classes são então, aos pares, aplicados ao exemplo consultado **x**, e produzem uma predição positiva para uma classe particular. Todos os votos dos $q(q-1)/2$ classificadores são marcados e a classe com mais votos representa a predição final. Portanto, os falsos positivos são eliminados neste segundo passo.

Observe que, no passo de eliminação de falsos positivos, a fronteira de decisão é desenhada entre duas classes verdadeiras de treinamento, ao invés de uma classe verdadeira e todas as outras classes complementares, o que seria mais complexo. Assim os falsos positivos são eliminados de modo mais eficaz. De acordo com DING & DUBCHAK (2001), o método Um contra Todos Único tem alta precisão de classificação.

O passo de eliminação do falso positivo é essencialmente uma técnica de redução de ruído. DING & DUBCHAK (2001) reduziram a razão do erro de uma rede neural por um fator de aproximadamente 2 nos experimentos realizados. Um problema com esta estratégia surge quando as classes recebem o mesmo número de votos, sendo que algum procedimento de decisão precisa ser adotado para tratar o empate.

4.10.3 Um contra Um

O método Um contra Um foi introduzido por KNERR (1990), e o primeiro a usar esta estratégia para SVM foram FRIEDMAN (1996) e KREBEL (1999). Este método constrói $K(K-1)/2$ classificadores, onde cada classificador é treinado sobre os dados de duas

classes. Para dados de treinamento envolvendo as *i*-ésimas e *j*-ésimas classes, soluciona-se o seguinte problema de classificação binária:

$$\min_{w^{ij},b^{ij},\xi^{ij}} \quad \frac{1}{2}(\mathbf{w}^{ij})^T\mathbf{w}^{ij} + C\sum_{t=1}^{N}\xi_t^{ij} \tag{4.107}$$

$$\begin{cases} (\mathbf{w}^{ij})^T\varphi(\mathbf{x}_t) + b^{ij} \geq 1-\xi_t^{ij}, & se \quad y_t = i, \\ (\mathbf{w}^{ij})^T\varphi(\mathbf{x}_t) + b^{ij} \leq 1-\xi_t^{ij}, & se \quad y_t = j, \\ \xi_t^{ij} \geq 0\,,\ t = 1,\cdots,N \end{cases}$$

Há diferentes procedimentos que permitem combinar estes classificadores visando atribuir uma única classe a um novo exemplo **x**, não pertencente ao conjunto de treinamento, depois que todos os $K(K-1)/2$ classificadores são construídos.

FRIEDMAN (1996) sugeriu usar a estratégia de votação: se $sign\left((\mathbf{w}^{ij})^T\varphi(\mathbf{x}) + b^{ij}\right)$ diz que **x** está na *i*-ésima classe, então a *i*-ésima classe recebe um voto. Caso contrário, a *j*-ésima classe recebe um voto. Então, prediz-se que **x** está na classe com maior número de votos. A abordagem de votação descrita é também chamada de estratégia *Max Wins*.

No caso de duas classes receberem o mesmo número de votos, uma estratégia de desempate deve ser adotada. HSU & LIN (2002) simplesmente selecionam aquela com menor índice. No entanto, é claro que tal estratégia não é adequada, uma vez que não existe nenhuma justificativa teórica e depende da implementação das classes. Na literatura, existem algumas estratégias para tratar deste problema.

Na prática, soluciona-se o problema dual associado ao problema definido pela expressão (4.107), cujo número de variáveis é igual ao número de dados das duas classes consideradas. Assim, se em média cada classe tem *N/K* amostras, então é necessário solucionar $K(K-1)/2$ problemas de programação quadrática, onde cada um deles tem, em média, 2*N/K* variáveis.

4.10.4 O método DAGSVM

O terceiro método a ser apresentado é o DAGSVM (do inglês *Directed Acyclic Graph Support Vector Machine*) proposto por PLATT *et al.* (1998). A fase de treinamento é

a mesma do método *Um contra Um*, requerendo a solução de $K(K-1)/2$ problemas de classificaçaõ binária. Entretanto, na fase de teste, é empregado um grafo direcionado acíclico com raiz binária que tem $K(K-1)/2$ nós internos e K folhas (veja Figura 4.22). Cada nó é um SVM binário envolvendo as *i*-ésima e *j*-ésima classes (representado por $f_{ij}(\mathbf{x})$). Dado um exemplo de teste **x**, partindo do nó raiz, a função decisão binária é avaliada. Então, move-se para a esquerda ou para a direita, dependendo do valor da saída. Portanto, caminha-se através de uma das alternativas, até encontrar um nó terminal que indica a classe.

Uma vantagem do DAG é que algumas análises de generalização podem ser estabelecidas (PLATT *et al.*, 1998). Além disso, o tempo de teste é menor que o tempo para o método Um contra Um, pois não é preciso executar os $K(K-1)/2$ classificadores. Não há ainda resultados teóricos similares para os métodos um contra um e um contra todos.

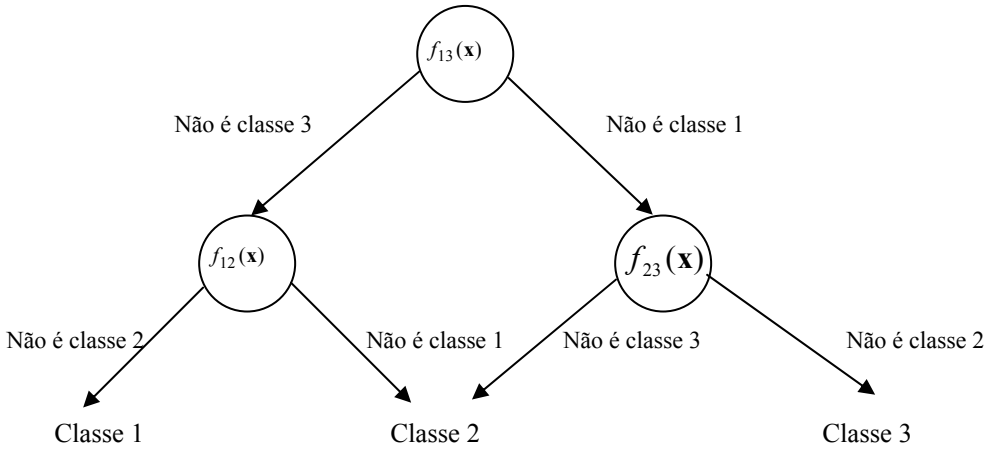


Figura 4.22 – Ilustração de um grafo direcionado acíclico para um problema com 3 classes no método DAGSVM.

4.10.5 Métodos que consideram todos os dados

VAPNIK (1998) e WESTON & WATKINS (1999) propuseram uma abordagem para problemas com múltiplas classes solucionando um único problema de otimização. A idéia é

similar à abordagem Um contra Todos. Constrói-se K regras entre duas classes, onde a j-ésima função $\mathbf{w}_{\mathrm{j}}^{T}\varphi(\mathbf{x})+b_{\mathrm{j}}$ separa os vetores de treinamento da classe j dos outros vetores, isto é, separa a classe j das outras classes. Assim, há K funções de decisão, mas todas são obtidas solucionando um único problema. A formulação é dada por:

$$\min_{\mathbf{w},b,\xi} \ \frac{1}{2}\sum_{j=1}^{K}(\mathbf{w}^{j})^{T}\mathbf{w}_{\mathrm{j}}+C\sum_{i=1}^{N}\sum_{j\neq y_i}^{K}\xi_i^{j} \tag{4.108}$$

$$\begin{cases}(\mathbf{w}^{y_i})^{T}\varphi(\mathbf{x}_i)+b_{y_i}\geq(\mathbf{w}^{j})^{T}\varphi(\mathbf{x}_i)+b^{j}+2-\xi_i^{j},\\ \xi_i^{j}\geq 0\\ i=1,\cdots,N\\ j\in\{1,\ldots,K\}\setminus y_i\end{cases}$$

Ao invés de trabalhar sobre o problema definido em (4.108), HSU & LIN (2002) adicionaram o termo $\sum_{j=1}^{K}(b^{j})^{2}$ à função objetivo, produzindo:

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\sum_{j=1}^{k}\left[(\mathbf{w}^{j})^{T} \quad b^{j}\right]\begin{bmatrix}\mathbf{w}^{j}\\ b^{j}\end{bmatrix}+C\sum_{i=1}^{N}\sum_{j\neq y_i}^{K}\xi_i^{j} \tag{4.109}$$

$$\begin{cases}\left[(\mathbf{w}^{y_i})^{T} \quad b^{y_i}\right]\begin{bmatrix}\varphi(\mathbf{x}_i)\\ 1\end{bmatrix}\geq\left[(\mathbf{w}^{j})^{T} \quad b^{j}\right]\begin{bmatrix}\varphi(\mathbf{x}_i)\\ 1\end{bmatrix}+2-\xi_i^{j}\\ \xi_i^{j}\geq 0\,,\ i=1,\cdots,N\,,\ j\in\{1,\ldots,K\}\setminus y_i\end{cases}$$

A função de decisão é dada por:

$$\arg\max_{j=1,\ldots,K}\ ((\mathbf{w}^{j})^{T}\varphi(\mathbf{x})+b^{j}),$$

que é a mesma daquela definida pelo método Um contra Todos, via expressão (4.104). Como no SVM binário, é mais fácil solucionar o problema dual associado ao problema definido pela expressão (4.108). Segundo WESTON & WATKINS (1999), a formulação dual é dada por:

$$\min_{\alpha} \ \sum_{i,k=1}^{N}\left(\frac{1}{2}c^{y_i}A_iA_k-\sum_{j=1}^{K}\alpha_i^{j}\alpha_k^{y_i}+\frac{1}{2}\sum_{j=1}^{K}\alpha_i^{j}\alpha_k^{j}\right)K(\mathbf{x}_i,\mathbf{x}_k)-2\sum_{i=1}^{N}\sum_{j=1}^{K}\alpha_i^{j} \tag{4.110}$$

sujeito a:

$$\begin{cases} \sum_{i=1}^{N} \alpha_i^j = \sum_{i=1}^{N} c_i^j A_i \,, \; j = 1,\ldots,K & (4.111) \\ 0 \le \alpha_i^j \le C, \, \alpha_i^{y_i} = 0, & (4.112) \\ A_i = \sum_{j=1}^{K} \alpha_i^j \,, \; c_k^{y_i} = \begin{cases} 1 & se \;\; y_i = y_k \\ 0 & se \;\; y_i \neq y_k \end{cases} & (4.113) \\ i = 1,\ldots,N \,, \; j = 1,\ldots,K \end{cases}$$

onde $K(\mathbf{x}_i, \mathbf{x}_j) = \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j)$. Então resulta:

$$\mathbf{w}_j = \sum_{i=1}^{N} (c_i^j A_i - \alpha_i^j) \varphi(\mathbf{x}_i), \;\; j = 1,\ldots,K \,, \tag{4.114}$$

e a função de decisão é dada na forma:

$$\arg\max_{j=1,\ldots,K} \left(\sum_{i=1}^{N} (c_i^j A_i - \alpha_i^j) K(\mathbf{x}, \mathbf{x}_i) + b_j \right)$$

A formulação apresentada na equação (4.108) é equivalente a duas outras formulações, a saber: formulação proposta por GUERMEUR (2000), apresentada abaixo:

$$\min_{\mathbf{w},b,\xi} \;\; \frac{1}{2} \sum_{o<j}^{K} \left\| \mathbf{w}^j - \mathbf{w}^o \right\| + C \sum_{i=1}^{N} \sum_{j \neq y_i}^{K} \xi_i^j \tag{4.115}$$

$$\begin{cases} (\mathbf{w}^{y_i})^T \varphi(\mathbf{x}_i) + b^{y_i} \ge (\mathbf{w}^j)^T \varphi(\mathbf{x}_i) + b^j + 2 - \xi_i^j \,, & (4.116) \\ \sum_{j=1}^{K} \mathbf{w}^j = 0, & (4.117) \\ \xi_i^j \ge 0 \,, \; i = 1,\cdots,N \,, \; j \in \{1,\ldots,k\} \,; \end{cases}$$

e formulação proposta por BREDNSTEINER & BENNETT (1999), definida como segue:

$$\min_{\mathbf{w},b,\xi} \;\; \frac{1}{2} \sum_{o<j}^{K} \left\| \mathbf{w}^j - \mathbf{w}^o \right\| + \sum_{j=1}^{K} (\mathbf{w}^j)^T \mathbf{w}_j + C \sum_{i=1}^{N} \sum_{j \neq y_i}^{K} \xi_i^j \,, \tag{4.118}$$

com as mesmas restrições do problema definido pela expressão (4.108)

Combinando as expressões (4.114) e (4.113), tem-se:

$$\sum_{j=1}^{K} \mathbf{w}^j = \sum_{j=1}^{K} \sum_{i=1}^{N} (c_i^j A_i - \alpha_i^j) \varphi(\mathbf{x}_i) = \sum_{i=1}^{N} (A_i - \sum_{j=1}^{K} \alpha_i^j) \varphi(\mathbf{x}_i) = 0 \,, \tag{4.119}$$

Assim, adicionando a restrição (4.117) ao problema definido pela expressão (4.108) não afeta a solução ótima. Então, a restrição (4.117) implica que:

$$\sum_{o<j}^{K} \left\| \mathbf{w}^j - \mathbf{w}^o \right\| = K \sum_{j=1}^{K} (\mathbf{w}^j)^T \mathbf{w}^j \,, \tag{4.120}$$

Logo, o problema definido pela expressão (4.108) e aquele definido pelas equações (4.115), (4.116) e (4.117) têm a mesma solução ótima. Argumentos similares podem também mostrar a relação entre o problema definido pela expressão (4.108) e aquele definido pela expressão (4.118). A equivalência destas formulações foi primeiramente abordada em GUERMEUR (2000).

Observe que a formulação (4.110) envolve KN variáveis. Como N variáveis são sempre nulas, $(K-1)/N$ variáveis precisam ser definidas. Infelizmente, esta redução é ainda insuficiente para que a formulação (4.110) seja considerada tratável computacionalmente, pois o produto entre K e N tende a assumir valores elevados. Esta é a razão pela qual WESTON & WATKINS (1999) tratam somente problemas de dimensão reduzida. HSU & LIN (2002) investigaram uma possível implementação do problema na formulação (4.110) para tratar problemas de dimensões maiores. No entanto, tal formulação ainda apresenta limitações em termos de tempo computacional.

4.10.6 Método de Crammer e Singer

CRAMMER & SINGER (2000) propuseram uma abordagem para problemas com múltiplas classes solucionando um único problema de otimização. Basicamente, o seguinte problema primal é tratado:

$$\min_{\mathbf{w}_m, \xi_i} \quad \frac{1}{2} \sum_{j=1}^{K} (\mathbf{w}^j)^T \mathbf{w}^j + C \sum_{i=1}^{N} \xi_i$$

sujeito a:

$$(\mathbf{w}^{y_i})_{y_i}^T \varphi(\mathbf{x}_i) - (\mathbf{w}^j) \varphi(x_i) \geq e_i^j - \xi_i \,,\ i = 1, \ldots, N \,, \tag{4.121}$$

onde $e_i^j = 1 - \delta_{y_i, j}$ e

$$\delta_{y_i,j} = \begin{cases} 1 & se \quad y_i = j \\ 0 & se \quad y_i \neq j \end{cases}.$$

Então a função de decisão é dada por:

$$\arg\max_{j=1,\ldots,K} \ \mathbf{w}_j^T \varphi(\mathbf{x})$$

A principal diferença entre a formulação (4.108) e a proposta em (4.121) é que a formulação (4.121) usa somente N variáveis de folga ξ_i, $i = 1,\ldots,N$. Isto é, ao invés de usar todo ξ_i^j como indicador de folga entre dois planos de decisão, esta proposta adota o máximo dentre K valores calculados, ou seja:

$$\xi_i = \left(\max_j \ ((\mathbf{w}^j)^T \varphi(\mathbf{x}) + e_i^j) - (\mathbf{w}^{y_i})^T \varphi(\mathbf{x}) \right)_+ ,$$

onde $(\cdot)_+ \equiv \max(\cdot,0)$. Além disso, a formulação (4.121) não contém coeficientes b_i, $i = 1,\ldots,N$. Observe que aqui não são consideradas diretamente as restrições inferiores $\xi_i \geq 0$, as quais podem ser obtidas quando $y_i = j$ e $e_i^j = 0$, produzindo $0 \geq 0 - \xi_i$ (equação (4.121)), ou seja, $\xi_i \geq 0$.

O problema dual associado à formulação (4.121) assume a forma:

$$\min_{\alpha} \ f(\alpha) = \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{N} K(\mathbf{x}_i, \mathbf{x}_k) \overline{\alpha}_i^T \overline{\alpha}_k + \sum_{i=1}^{N} \overline{\alpha}_i^T \overline{e}_i \tag{4.122}$$

$$\begin{cases} \displaystyle\sum_{j=1}^{K} \alpha_i^j = 0, \ i = 1,\ldots,N, & (4.123) \\ \alpha_i^j \leq 0, \ se \ \ y_i \neq j, & (4.124) \\ \alpha_i^j \leq C, \ se \ \ y_i = j, & \\ i = 1,\ldots,N, \ j = 1,\ldots,K & \end{cases}$$

onde $K(\mathbf{x}_i, \mathbf{x}_k) = \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_k)$, $\overline{\alpha}_i = [\alpha_i^1,\ldots,\alpha_i^K]^T$ e $\overline{e}_i = [e_i^1,\ldots,e_i^K]^T$. Então:

$$\mathbf{w}^j = \sum_{i=1}^{N} \alpha_i^j \varphi(\mathbf{x}_i)$$

Expressando α e e tal que $\boldsymbol{\alpha} \equiv [\alpha_1^1,\ldots,\alpha_1^K,\ldots,\alpha_N^1,\ldots,\alpha_N^K]^T$ e $\mathbf{e} \equiv [e_1^1,\ldots,e_1^K,\ldots,e_N^1,\ldots,e_N^K]^T$, então a função-objetivo dual pode ser escrita como:

$$\frac{1}{2}\boldsymbol{\alpha}^T(\mathbf{K}\otimes\mathbf{I})\boldsymbol{\alpha}+\mathbf{e}^T\boldsymbol{\alpha}\,, \tag{4.125}$$

onde $\mathbf{I}$ é uma matriz identidade de dimensão $K\times K$ e $\otimes$ é o produto de Kronecher. Uma vez que $\mathbf{K}$ é semi-definida positiva, $\mathbf{K}\otimes\mathbf{I}$ é também semi-definida positiva. Este é uma outra forma de indicar que a formulação (4.122) apresenta um problema de otimização convexo.

A função de decisão é dada por:

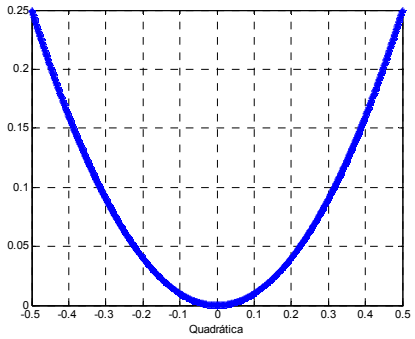
$$\arg\max_{j=1,\ldots,K}\sum_{i=1}^{N}\alpha_i^j K(\mathbf{x},\mathbf{x}_i)$$

A principal diferença entre as restrições lineares das formulações (4.111) e (4.123) é que em (4.111) temos K equações, enquanto que em (4.123) temos N equações. É interessante observar que estas equações vêm das condições KKT sobre diferentes variáveis do problema primal. A formulação (4.111) envolve as variáveis irrestritas $b_1,\ldots,b_K$ em (4.108), enquanto que a formulação (4.123) envolve as variáveis irrestritas $\xi_1,\ldots,\xi_N$. Logo a formulação (4.123) é mais simples que a formulação (4.111), uma vez que cada uma de suas N equações independentes envolvem apenas K variáveis. Conclui-se então que, com base na formulação (4.123), o tratamento de problemas com um número grande de amostras de treinamento fica mais fácil de ser tratado (HSU & LIN, 2002).

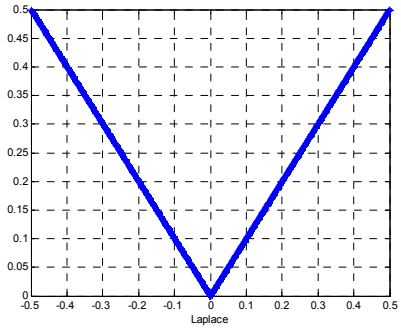
4.11 Máquinas de vetores-suporte para regressão

A extensão de máquinas de vetores-suporte para o tratamento de problemas de regressão se dá pela introdução de uma função de perda (SMOLA, 1996). Em linhas gerais, vai ocorrer uma reversão de propósito quando comparado ao tratamento adotado junto a problemas de classificação binária, ou seja, em lugar de forçar as amostras a se posicionarem o mais distante possível do hiperplano ótimo, no espaço de características, o propósito agora é forçar as amostras a se posicionarem o mais próximas possível do hiperplano ótimo, não importa a que lado do hiperplano a amostra se encontre. Embora os propósitos da otimização sejam opostos, o objetivo de ambas as formulações é o mesmo:

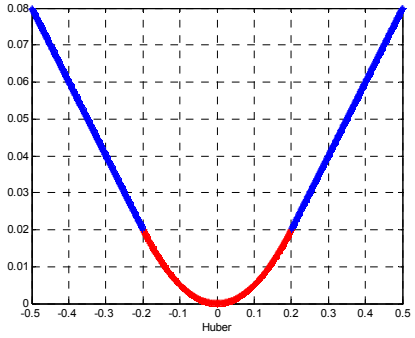
maximizar a capacidade de generalização. A Figura 4.23 ilustra cinco possíveis tipos de funções de perda.



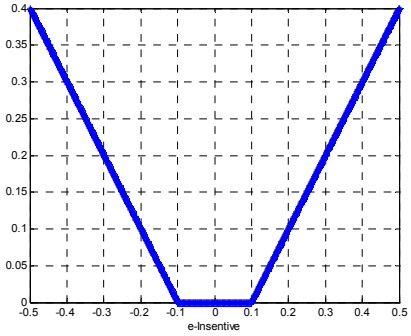
(a) Quadrática



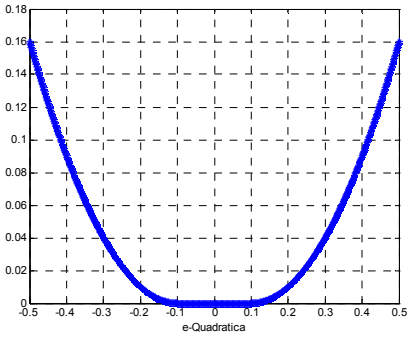
(b) Laplace



(c) Huber



(d) ε - insensível



(e) ε-Quadrática

Figura 4.23 – Função de perda para problemas de regressão.

A função de perda mostrada na Figura 4.23(a) corresponde ao critério convencional de minimização do erro quadrático. A função de perda mostrada na Figura 4.23(b) é uma

função de perda que é menos sensível a *outliers* que a função de perda quadrática pela inclinação adotada para as duas semiretas e supondo que os outliers não vão se distanciar tanto do hiperplano ótimo. HUBER (1981) propôs a função de perda mostrada na Figura 4.23(c) como uma função de perda que tem propriedades desejáveis quando a distribuição dos dados é desconhecida. Estas três funções geralmente não produzem vetores-suporte esparsos. Para tanto, VAPNIK (1995) propôs as funções de perda mostradas nas Figuras 4.23(d) e (e), como aproximações para as funções de perda de Laplace e Huber, respectivamente.

Repare que estas duas propostas de funções de perda, Figuras 4.23(d) e (e), não penalizam as amostras que se encontram até uma distância ε do hiperplano ótimo, no espaço de características.

4.12 Regressão linear

Considere o problema de aproximar um conjunto de dados de treinamento, composto pelas seguintes amostras de entrada-saída:

$$(\mathbf{x}_1, y_1),...,(\mathbf{x}_N, y_N), \mathbf{x} \in \Re^n, y \in \Re,$$

com uma função linear

$$f(\mathbf{x}) = (\mathbf{w} \cdot \mathbf{x}) + b \tag{4.126}$$

A função de regressão é dada pelo mínimo do funcional,

$$\Phi(\mathbf{w}, \xi^* \xi) = \frac{1}{2}(\mathbf{w}^T \mathbf{w}) + C(\sum_{i=1}^{N} \xi + \sum_{i=1}^{N} \xi_i^*), \tag{4.127}$$

onde C é um valor especificado pelo usuário, ξ e ξ^* são variáveis de folga representando as restrições dos dois lados do hiperplano, conforme as equações abaixo:

$$\begin{cases} y_i - (\mathbf{w} \cdot \mathbf{x}_i) - b \le \varepsilon + \xi_i \\ (\mathbf{w} \cdot \mathbf{x}_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0, \quad i = 1, \ldots, N \end{cases} \tag{4.128}$$

A Figura 4.24 ilustra graficamente esta situação para uma SVM com *kernel* linear e função de perda ε-insensível, onde somente as amostras fora da região sombreada contribuem para

o funcional da equação (4.127), isto é, as variações que excedam ε são penalizadas, neste caso, de forma linear.

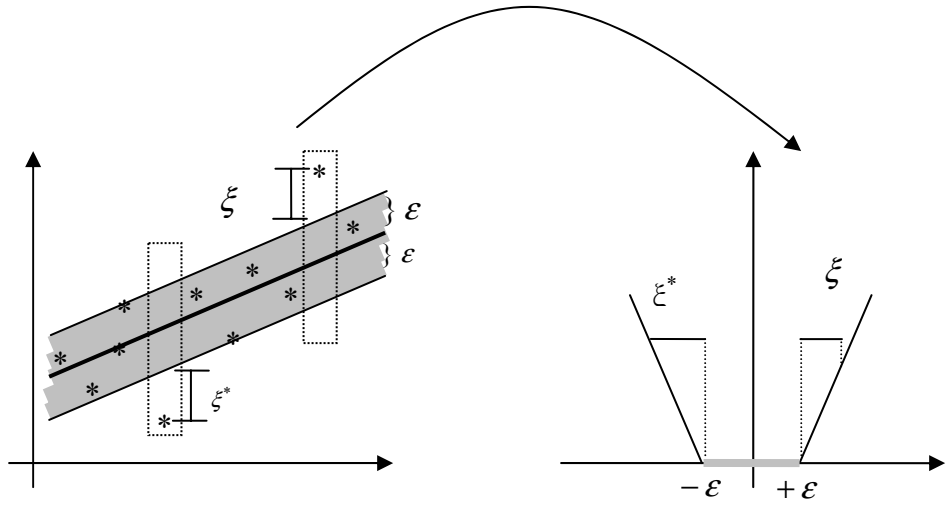


Figura 4.24 – Ilustração do processo de penalização da função de perda para uma SVM com *kernel* linear e função de perda ε-insensível.

4.12.1 Função de perda ε-insensível

Usando uma função de perda ε-insensível, apresentada na Figura 4.23(d), cuja equação é dada por:

$$L_\varepsilon = \begin{cases} 0 & se \quad |f(\mathbf{x}) - y| < \varepsilon \\ |f(\mathbf{x}) - y| - \varepsilon & \text{caso contrário} \end{cases},$$

a formulação assume a seguinte configuração:

$$\max_{\alpha,\alpha^*} \left[-\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^{N} \alpha_i^* (y_i - \varepsilon) - \alpha_i (y_i + \varepsilon) \right] \qquad (4.129)$$

ou, de forma alternativa:

$$\max_{\alpha,\alpha^*} \left[-\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) y_i + \sum_{i=1}^{N} (\alpha_i + \alpha_i^*) \varepsilon \right] \qquad (4.130)$$

sujeito a:

$$\begin{cases} 0 \le \alpha_i \le C \quad i = 1,...,N \\ 0 \le \alpha_i^* \le C \quad i = 1,...,N \\ \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0 \end{cases} \tag{4.131}$$

Solucionando o problema vinculado à formulação (4.129), com as restrições definidas em (4.131), podemos determina os multiplicadores de Lagrange, α, α^*, e a função de regressão é dada pela equação (4.126), onde

$$\mathbf{w} = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \mathbf{x}_i \tag{4.132}$$

$$b = -\frac{1}{2} \left((\mathbf{w} \cdot \mathbf{x}_r) + (\mathbf{w} \cdot \mathbf{x}_s) \right) \tag{4.133}$$

As condições de Karush-Khun-Tucker (KKT) que são satisfeitas pela solução são:

$$\alpha_i \alpha_i^* = 0 \quad i = 1,...,N \tag{4.134}$$

Portanto, os vetores-suporte vão corresponder às amostras para as quais exatamente um dos multiplicadores de Lagrange é maior que zero. Quando $\varepsilon = 0$ (Figura 4.23(b)), obtém-se a função de perda de Laplace, e o problema de otimização é simplificado, adotando $\beta_i = \alpha_i - \alpha_i^*$, conduzindo à seguinte formulação:

$$\min_{\beta} \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \beta_i \beta_j (\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^{N} \beta_i y_i \tag{4.135}$$

sujeito a:

$$-C \le \beta_i \le C, i = 1,...,N, \tag{4.136}$$

$$\sum_{i=1}^{N} \beta_i = 0. \tag{4.137}$$

A função de regressão é dada pela equação (4.126), onde

$$\mathbf{w} = \sum_{i=1}^{N} \beta_i x_i, \tag{4.138}$$

$$b = -\frac{1}{2} \left((\mathbf{w} \cdot \mathbf{x}_r) + (\mathbf{w} \cdot \mathbf{x}_s) \right). \tag{4.139}$$

4.12.2 Função de perda ε-Quadrática

Usando a função de perda ε-quadrática, Figura 4.23(e), dada por:

$$L_{quad}(f(\mathbf{x})-y)=\begin{cases}0 & se\ \ |f(\mathbf{x})-y| \\ (f(\mathbf{x})-y)^2-\varepsilon & \text{caso contrário}\end{cases},$$

a formulação assume a seguinte configuração:

$$\max_{\alpha,\alpha^*}\left[-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\mathbf{x}_i\cdot\mathbf{x}_j)+\sum_{i=1}^{N}\alpha_i^*(y_i-\varepsilon)-\alpha_i(y_i+\varepsilon)\right] -\frac{1}{2C}\sum_{i=1}^{N}(\alpha_i^2+(\alpha_i^*)^2) \tag{4.140}$$

sujeito a:

$$\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)=0\,. \tag{4.141}$$

A função de regressão é dada pelas equações (4.126) e (4.139).

Quando ε = 0 (Figura 4.23(a)), obtém-se a função de perda quadrática e o problema de otimização correspondente pode escrito como:

$$\max_{\alpha,\alpha^*}\left[-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)(\mathbf{x}_i\cdot\mathbf{x}_j)+\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)y_i-\frac{1}{2C}\sum_{i=1}^{N}(\alpha_i^2+(\alpha_i^*)^2)\right] \tag{4.142}$$

sujeito a:

$$\sum_{i=1}^{N}(\alpha_i^*-\alpha_i)=0\,. \tag{4.143}$$

Fazendo a seguinte substituição $\beta_i=\alpha_i-\alpha_i^*$ junto à formulação (4.142), viabiliza-se uma simplificação via condições de KKT (equação (4.134)), o que implica $\beta_i^2=\alpha_i^2+(\alpha_i^*)^2$. O problema de otimização resultante é dado por:

$$\min_{\beta}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j(\mathbf{x}_i\cdot\mathbf{x}_j)-\sum_{i=1}^{N}\beta_i y_i+\frac{1}{2C}\sum_{i=1}^{N}\beta_i^2\,, \tag{4.144}$$

sujeito a:

$$\sum_{i=1}^{N}\beta_i=0\,. \tag{4.145}$$

A função de regressão é dada pela equação (4.126), onde:

$$\mathbf{w} = \sum_{i=1}^{N} \beta_i \mathbf{x}_i \quad . \tag{4.146}$$

$$b = -\frac{1}{2}\left((\mathbf{w} \cdot \mathbf{x}_r) + (\mathbf{w} \cdot \mathbf{x}_s)\right). \tag{4.147}$$

4.12.3 Função de perda de Huber

Usando a função de perda quadrática, Figura 4.23(c), na forma:

$$L_{Huber}\ (f(\mathbf{x}) - y) = \begin{cases} \frac{1}{2}(f(\mathbf{x}) - y)^2 & se \quad |f(\mathbf{x}) - y| < c \\ c|f(\mathbf{x}) - y| - \frac{c^2}{2} & \text{caso contrário} \end{cases} \tag{4.148}$$

a formulação assume a seguinte configuração:

$$\max_{\alpha,\alpha^*}\left[-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)(\mathbf{x}_i \cdot \mathbf{x}_j) + \sum_{i=1}^{N}(\alpha_i^* - \alpha_i)y_i - \frac{c}{2C}\sum_{i=1}^{N}(\alpha_i^2 + (\alpha_i^*)^2) \right] \tag{4.149}$$

O problema de otimização resultante é dado por:

$$\min_{\beta} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\beta_i\beta_j(\mathbf{x}_i \cdot \mathbf{x}_j) - \sum_{i=1}^{N}\beta_i y_i + \frac{c}{2C}\sum_{i=1}^{N}\beta_i^2 \tag{4.150}$$

sujeito a:

$$\begin{cases} -C \le \beta_i \le C, i = 1,...,N \\ \sum_{i=1}^{N}\beta_i = 0 \end{cases} \tag{4.151}$$

e a função de regressão é dada pelas equações (4.126) , (4.139) e (4.149).

4.12.4 Exemplo de aplicação

Considere o conjunto de dados apresentados na Tabela 4.3. A solução SVM para a função de perda da Figura 4.23(b), com nenhum controle da capacidade de modelagem, isto é, com $C \to \infty$, é mostrada na Figura 4.25.

x	y
1,0	-1,6
3,0	-1,8
4,0	-1,0
5,6	1,2
7,8	2,2
10,2	6,8
11,0	10,0
11,5	10,0
12,7	10,0

Tabela 4.3 – Dados para o problema de regressão

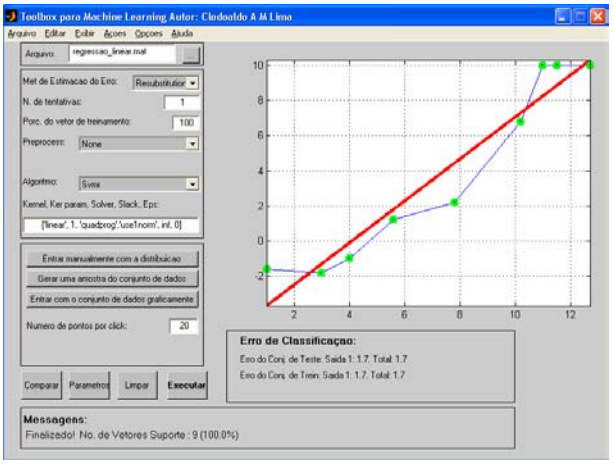


Figura 4.25 – Regressão Linear usando SVM.

4.13 Regressão não-linear

De forma similar aos problemas de classificação, um modelo não-linear é usualmente requerido para representar adequadamente o mapeamento no espaço original. Repare que este mapeamento será sempre um hiperplano no espaço de características. Da mesma

maneira que na abordagem SVM para classificação não-linear, um mapeamento não-linear pode ser empregado na definição do espaço de características de alta dimensionalidade, onde sempre uma regressão linear será realizada.

A abordagem baseada em funções *kernel* é novamente empregada para tratar a maldição da dimensionalidade. A formulação SVM, agora com *kernel* não-linear e tomando como função de perda ε-insensível aquela da Figura 4.23(d), é dada por:

$$\max_{\alpha,\alpha^*}\left[-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)K(\mathbf{x}_i,\mathbf{x}_j)+\sum_{i=1}^{N}\alpha_i^*(y_i-\varepsilon)-\alpha_i(y_i+\varepsilon)\right] \tag{4.152}$$

sujeito a:

$$\begin{cases} 0\le\alpha_i\le C \quad i=1,...,N \\ 0\le\alpha_i^*\le C \quad i=1,...,N \\ \sum_{i=1}^{N}(\alpha_i^*-\alpha_i)=0 \end{cases} \tag{4.153}$$

Resolvendo a equação (4.152) com as restrições em (4.153), obtêm-se os multiplicadores de Lagrange, α_i,α_i^*, e a função de regressão é dada por:

$$f(\mathbf{x})=\sum_{SVs}(\alpha_i-\alpha_i^*)K(\mathbf{x},\mathbf{x}_i)+b \tag{4.154}$$

onde

$$\begin{cases} (\mathbf{w}\cdot\mathbf{x})=\sum_{i=1}^{N}(\alpha_i-\alpha_i^*)K(\mathbf{x}_i,\mathbf{x}_j) \\ b=-\frac{1}{2}\sum_{i=1}^{N}(\alpha_i-\alpha_i^*)(K(\mathbf{x}_i,\mathbf{x}_r)+K(\mathbf{x}_i,\mathbf{x}_s)) \end{cases} \tag{4.155}$$

Da mesma forma que na formulação SVM para classificação, a restrição de igualdade pode ser abandonada se o *kernel* contiver um termo de bias *b* sendo fornecido dentro da função *kernel*. Sendo assim, a função de regressão é dada por:

$$f(\mathbf{x})=\sum_{SVs}(\alpha_i-\alpha_i^*)K(\mathbf{x},\mathbf{x}_i) \tag{4.156}$$

O critério de otimização para outras funções de *kernel* mostradas na Figura 4.23 podem ser obtidas de modo análogo, substituindo o produto interno por uma função *kernel*.

A função de perda ε-insensível não requer que todas as amostras sejam vetores-suporte, como ocorre quando a função custo é quadrática ou de Huber.

4.13.1 Exemplos didáticos

Para ilustrar alguns experimentos com função *kernel* não-linear, várias funções *kernel* foram usadas para modelar os dados de regressão da Tabela 4.3, com uma função de perda ε-insensível ($\varepsilon = 0{,}5$) e $C \to \infty$. A Figura 4.26 mostra a solução SVM para um polinômio de grau 2, com os vetores-suporte circulados em verde. A linha pontilhada descreve a região ε-insensível em torno da solução. Repare que, se todas as amostras residem dentro desta região, haverá erro zero associado à função de perda. Os resultados demonstram que não há vetores-suporte dentro da região ε-insensível.

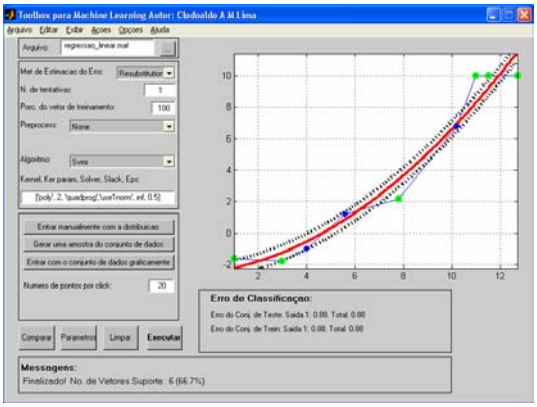


Figura 4.26 – Regressão com *kernel* do tipo polinomial

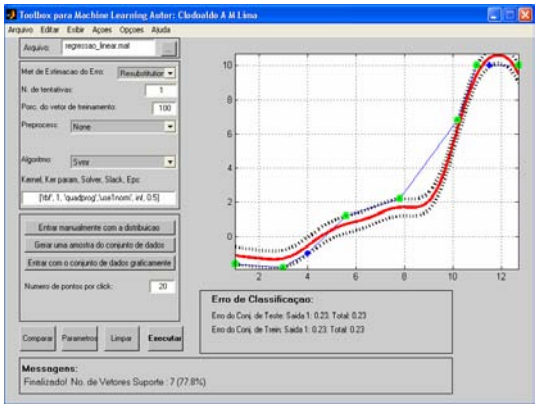


Figura 4.27 – Regressão com *kernel* do tipo função de base radial

A Figura 4.27 ilustra a solução SVM para função de base radial com $\sigma = 1{,}0$. Neste exemplo, o modelo é bastante flexível para modelar a função com erro zero associado à função de perda, como é verificado pelo fato de que todas as amostras residem sobre, ou dentro, da zona ε-insensível.

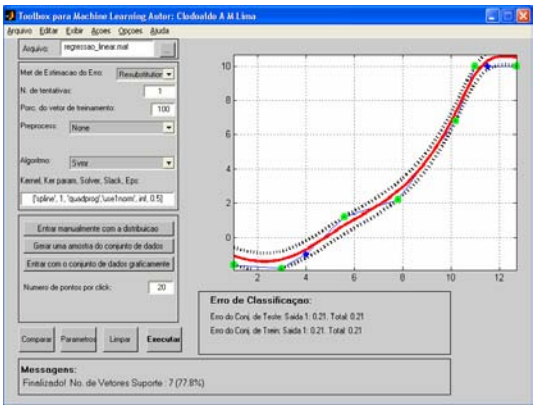


Figura 4.28 – Regressão com *kernel* do tipo spline linear.

A Figura 4.28 mostra a solução para *kernel* do tipo spline linear. O modelo resultante é um spline cúbico por partes e, novamente devido à elevada capacidade de mapeamento desta função, é possível obter uma solução com erro zero junto à função de perda. Além disso, a solução se mostrou mais suave que aquela apresentada na Figura 4.27.

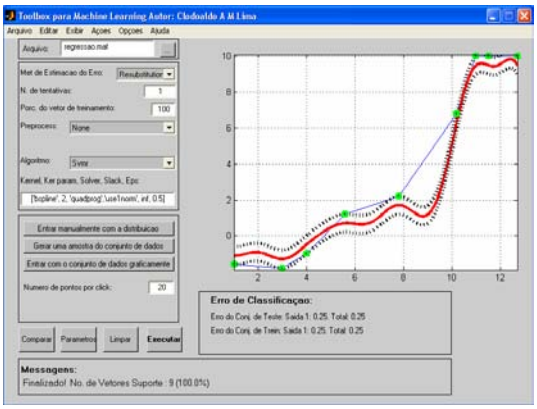


Figura 4.29 – Regressão com *kernel* do tipo B-spline infinito.

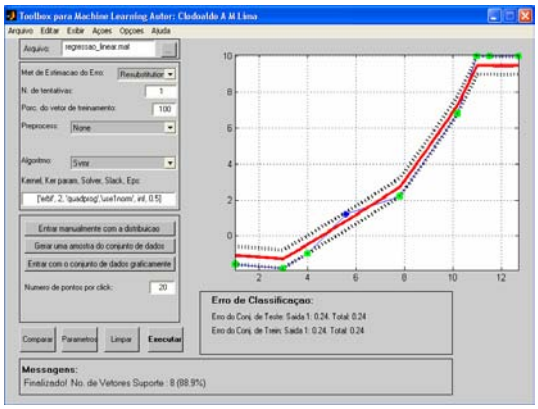


Figura 4.30 – Regressão com *kernel* do tipo exponencial RBF.

A Figura 4.29 mostra a solução para *kernel* B-spline infinito. Repare que todas as amostras foram tomadas como vetores-suporte, de modo que a capacidade de generalização do modelo tende a ser inferior à de outros modelos mais parcimoniosos. Já a Figura 4.30 mostra a solução para *kernel* exponencial RBF, que é um spline linear por partes. Embora

este modelo exiba alta capacidade de modelagem, ele apresentou um comportamento não desejado nas extremidades da região de aproximação.

4.13.2 Comentários

Dos resultados apresentados acima, pode-se extrair uma conclusão básica: para problemas de regressão, é necessário selecionar adequadamente tanto a função de perda quanto o parâmetro que controla a capacidade de modelagem. Estas considerações devem ser baseadas em conhecimento a priori acerca do problema e da distribuição do ruído. Na ausência desta informação, a função de perda de Huber, Figura 4.23(c), apresenta-se como uma boa alternativa (VAPNIK, 1995). VAPNIK (1995) propôs as duas funções de perda ε-insensívis, apresentadas nas Figuras 4.23(d) e (e), como um *trade-off* entre as funções de perda de Laplace e Huber e funções de perda que favorecem a redução do número de vetores-suporte. Entretanto, sua implementação é mais custosa computacionalmente e a região de ε-insensibilidade pode apresentar problemas, como será discutido na próxima seção.

4.13.3 Caso de Estudo: dados Titanium

O exemplo apresentado nesta seção considera os dados Titanium (DIERCKX, 1993) como um exemplo ilustrativo de um problema de regressão não-linear unidimensional. Há três formas de controlar o modelo de regressão: (1) via função de perda; (2) via função *kernel*; e (3) via controle da capacidade de modelagem, associada ao parâmetro *C*. Os resultados a serem apresentados a seguir foram obtidos usando uma função de perda ε-insensível ($\varepsilon = 0{,}05$), com diferentes *kernels* e diferentes graus de controle da capacidade de modelagem.

A Figura 4.31 ilustra o desemplenho da função *kernel* spline linear e nenhum controle da capacidade de modelagem. Como esperado, a solução reside dentro da região ε - insensível.

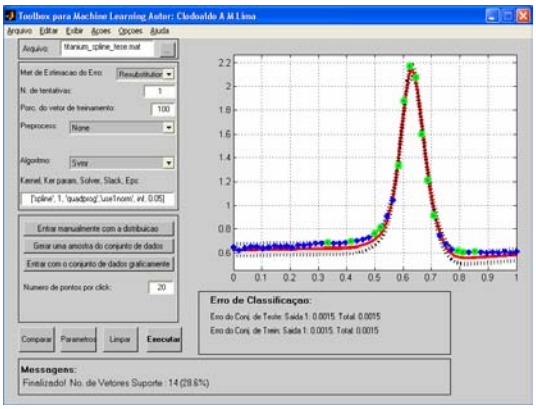


Figura 4. 31 – Regressão usando *kernel* Spline Linear para Titanium ($\varepsilon = 0{,}05$, $C \to \infty$).

A Figura 4.32 ilustra a solução para o *kernel* B-spline sem controle da capacidade de modelagem. O *kernel* B-spline, em particular, pode se mostrar propenso a oscilações quando a região ε-insensível é adotada. Logo, sugere-se a escolha do *kernel* spline linear, ou de uma função de perda alternativa.

A Figura 4.33 ilustra a solução para o *kernel* RBF ($\sigma = 1{,}0$), sem controle da capacidade de modelagem. Com este *kernel*, o problema de regressão não foi adequadamente resolvido.

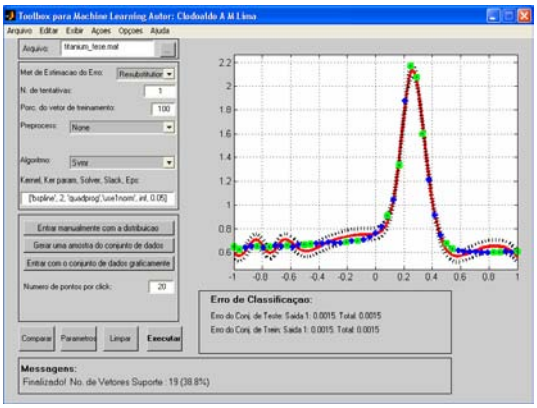


Figura 4.32 – Regressão usando B-spline para Titanium ($\varepsilon = 0{,}05$, $C \to \infty$).

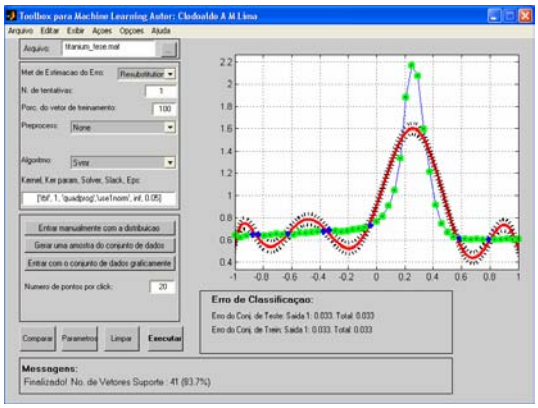


Figura 4.33 – Regressão usando RBF gaussiano para Titanium (ε=0,05, σ=1,0, $C \to \infty$).

A Figura 4.34 ilustra a solução para o *kernel* RBF gaussiano ($\sigma = 0{,}3$) sem controle da capacidade de modelagem. Pode ser visto que o *kernel* RBF é agora capaz de fornecer uma solução mais aceitável. Entretanto, passa a ocorrer oscilação, a qual não é penalizada na região de ε-insensibilidade.

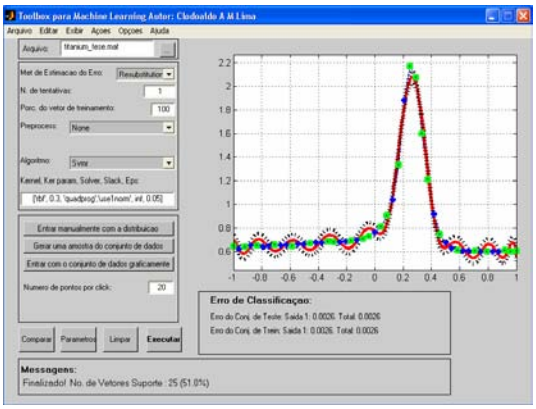


Figura 4.34 – Regressão usando RBF Gaussiano para Titanium (ε=0.05, σ = 0.03, $C \to \infty$).

A Figura 4.35 ilustra a solução para uma *kernel* RBF exponencial (σ =0,3) sem controle da capacidade de modelagem. A correspondente solução é uma função linear por partes e, conseqüentemente, a oscilação é evitada.

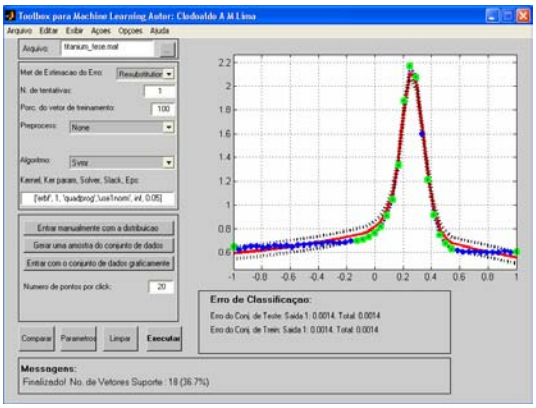


Figura 4.35 – Regressão usando RBF exponencial para Titanium (ε = 0,05, σ = 1,0, $C \to \infty$).

A Figura 4.36 ilustra a solução para *kernel* Fourier com grau 3, sem controle da capacidade de modelagem A solução sofre problemas similares àqueles obtidos com *kernel* RBF (veja Figura 4.33).

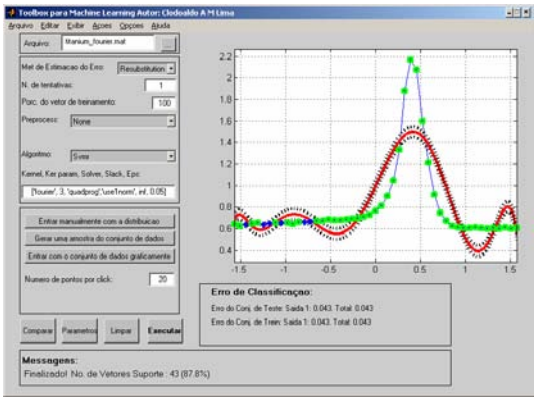


Figura 4.36 – Regressão usando Fourier para Titanium ($\varepsilon = 0{,}05$, grau 3, $C \to \infty$).

A Figura 4.37 ilustra a solução para *kernel* linear spline, com controle da capacidade de modelagem, tomando $C = 10$. O valor de c se mostra inconveniente, quando comparado com a solução obtida na Figura 4.31.

A Figura 4.38 ilustra a solução para um *kernel* Bspline, com controle da capacidade de modelagem, tomando $C = 10$. Novamente, o valor de c se mostra inconveniente, quando comparado com a solução obtida na Figura 4.32.

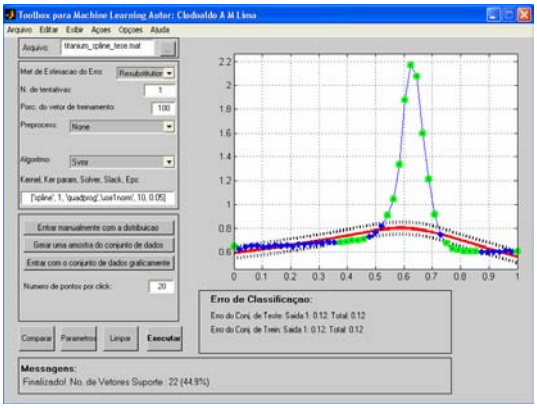


Figura 4.37 – Regressão usando Spline Linear para Titanium ($\varepsilon = 0{,}05$, $C = 10$).

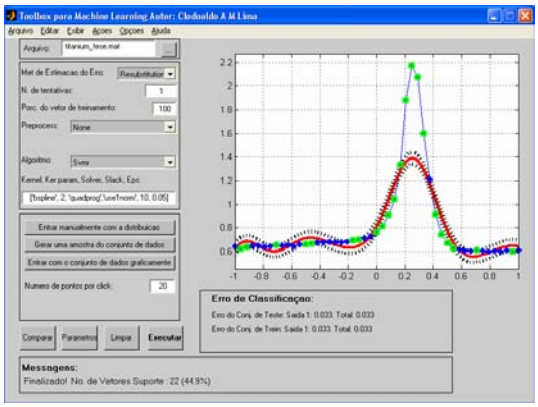


Figura 4.38 – Regressão usando B-spline para Titanium ($\varepsilon = 0{,}05$, $C = 10$).

Os exemplos que foram apresentados aqui não permitem formular conclusões genéricas, ou seja, válidas para outros cenários vinculados a problemas de regressão. Vale salientar que um valor de 0,05 para ε leva a uma região ε-insensível muito ampla. Em casos assim, uma seleção cuidadosa do parâmetro C e o emprego de técnicas estatísticas como validação cruzada devem ser considerados.

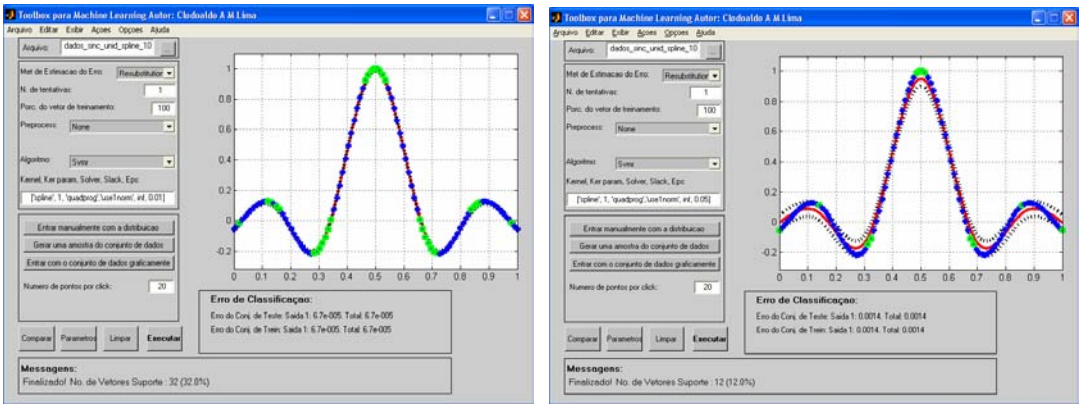
A ocorrência de oscilação restrita à faixa de ε-insensibilidade é um indicativo de que tudo é possível naquela região, até que se imponha algum critério adicional de desempenho. Alguns experimentos realizados no Capítulo 5 mostram que tais oscilações podem ser reduzidas utilizando abordagem baseada em um ensemble.

4.14 Aplicação de SVM à problemas de aproximação de funções

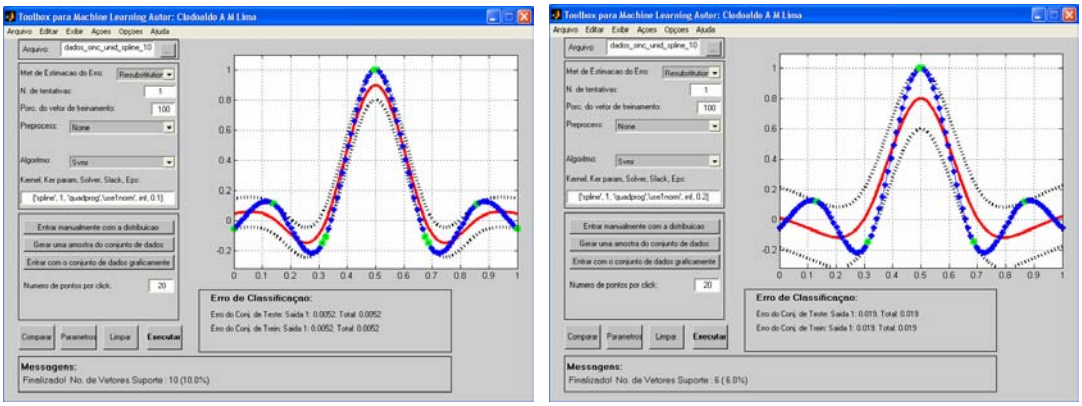
Esta seção tem como objetivo demonstrar que o número de vetores-suporte que são utilizados para construir o regressor depende da precisão desejada ε. Quanto menor a precisão de aproximação, isto é, quanto maior o valor de ε, menos vetores-suporte serão necessários. Com relação à capacidade de generalização esta também será afetada, podendo diminuir ou não com o aumento do valor de ε. Em todos os experimentos, foi utilizado o *kernel* spline, uma vez que este *kernel* não necessita da sintonia de qualquer parâmetro.

Primeiramente, serão descritos os experimentos para a função *sinc* unidimensional (veja Apêndice A) a qual foi amostrada uniformemente no intervalo [0,20]. O conjunto de dados de treinamento é composto de 100 amostras. Posteriormente, serão descritos experimentos envolvendo a função *sinc* bi-dimensional (veja Apêndice A) amostrada no intervalo $[0,20]^2$.

A Figura 4.39 mostra a aproximação da função *sinc* unidimensional para diferentes níveis de precisão, isto é, diferentes valores de ε usando a função de perda ε-insensível e o valor $C \to \infty$, onde os círculos em verde indicam os vetores-suporte. Pode-se observar que, decrementando a precisão da aproximação (que é equivalente a aumentar o valor de ε), o número de vetores-suporte diminui.



(a) (b)



(c) (d)

Figura 4.39 – Regressores SVM com *kernel* spline para diferentes níveis de precisão associados à função de perda ε-insensível: (a) 32 SVs para $\varepsilon = 0{,}01$, (b) 12 SVs para $\varepsilon = 0{,}05$, (c) 10 SVs para $\varepsilon = 0{,}1$, (d) 6 SVs para $\varepsilon = 0{,}2$.

Os resultados para a função de perda ε-quadrática e Huber apresentaram comportamento similar, quando adota-se $C \to \infty$. Desta forma, o valor de C foi alterado para $C = 10^4$ para as funções de perda ε-quadrática e Huber, com o intuito de visualizar a influência do parâmetro ε no número de vetores-suporte. Os resultados obtidos são apresentados nas Figuras 4.40 e 4.41. Comparando estes resultados, pode-se observar que a função de perda ε-quadrática utiliza um número maior de vetores-suporte em relação à função de perda de Huber, para a mesma precisão. Isto se deve à forma com a qual é realizada a penalização do erro. Enquanto na função de perda ε-quadrática o erro é elevado

ao quadrado, na função de Huber, caso o erro caia abaixo de um valor, este será considerado também elevado ao quadrado. Caso contrário, será idêntico à função de perda ε-insensível.

Foram realizadas também simulações com a função de perda ε-insensível para $C = 10^4$. Embora estes resultados não estão sendo apresentados, comparando os resultados obtidos com aqueles apresentados nas Figura 4.40 e 4.41 pode-se concluir que a função de perda ε-insensível precisa de menos vetores-suporte para produzir a mesma aproximação com a mesma precisão. Isto é esperado, conforme já mencionado, pois as funções de perda ε-quadrática e de Huber produzem uma aproximação menos esparsa, em termos do número de vetores suporte, quando comparadas com a função de perda ε-insensível.

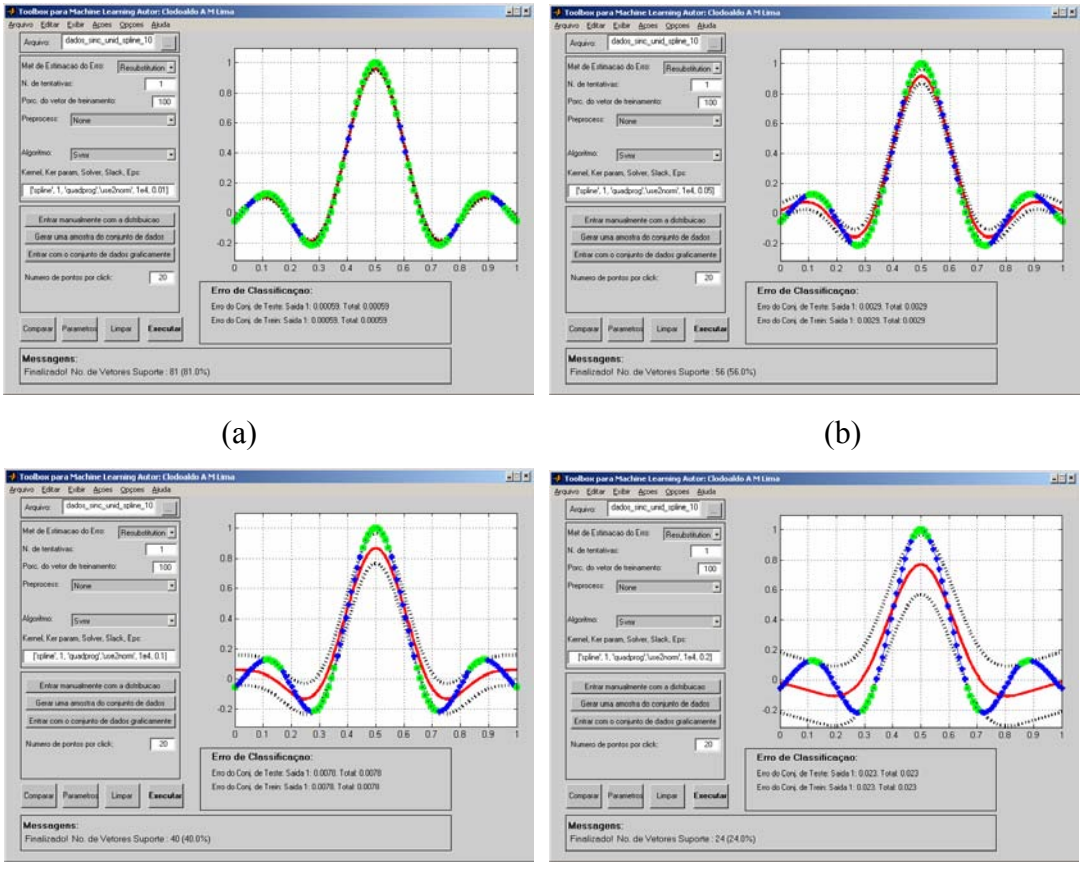
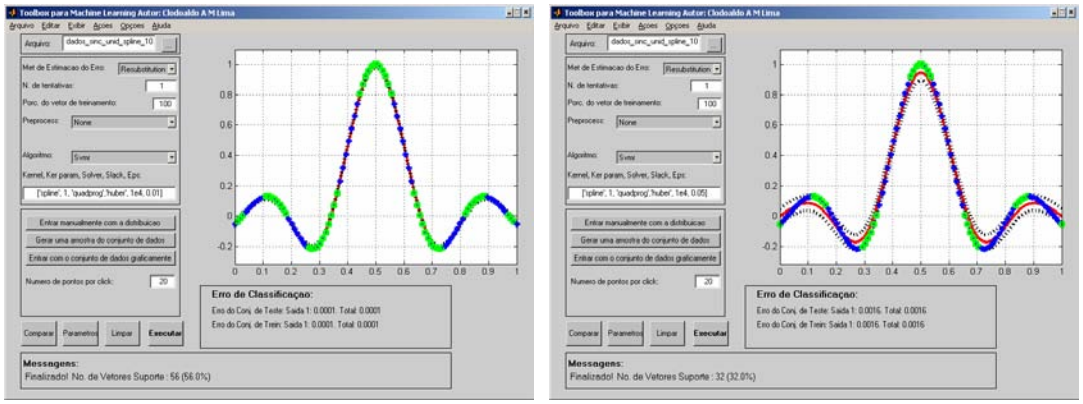
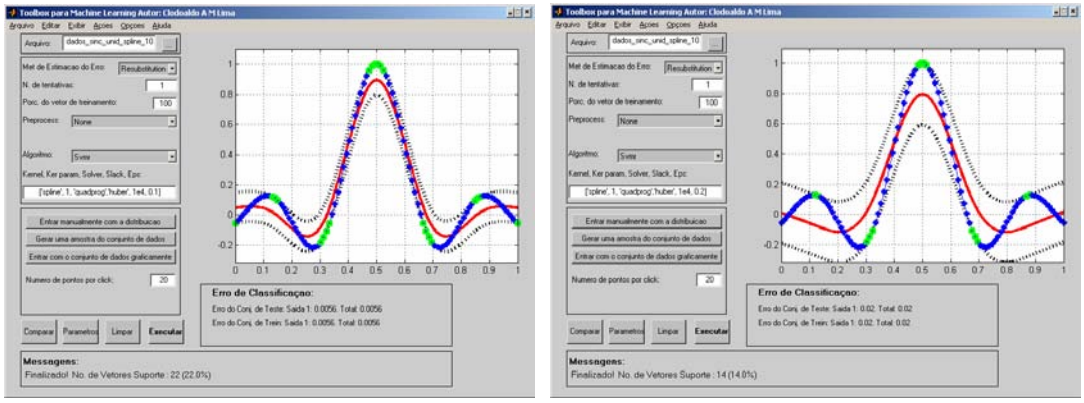


Figura 4.40 – Regressores SVM para diferentes níveis de precisão associados à função de perda ε-quadrática (C = 10^4): (a) 81 SVs para ε = 0,01, (b) 56 SVs para ε = 0,05, (c) 40 SVs para ε = 0,1, (d) 24 SVs para ε = 0,2.



(a) (b)



(c) (d)

Figura 4.41 – Regressores SVM para diferentes níveis de precisão associados à função de perda de Huber ($C = 10^4$): (a) 56 SVs para $\varepsilon = 0{,}01$, (b) 32 SVs para $\varepsilon = 0{,}05$, (c) 22 SVs para $\varepsilon = 0{,}1$, (d) 14 SVs para $\varepsilon = 0{,}2$.

A Figura 4.42 apresenta resultados obtidos para a função sinc bi-dimensional variando a quantidade de amostras mantendo a mesma precisão $\varepsilon = 0{,}03$. Pode-se observar que uma mudança no número de amostras de treinamento por um fator de 9 (225 para 2025), produz um mudança no número de vetores-suporte por um fator 2,2: 123 SVs na Figura 4.42(b), e 275 SVs na Figura 4.42(c). Estes resultados apontam para a natureza esparsa da solução baseada em SVM e uma relativa robustez à quantidade de dados amostrados.

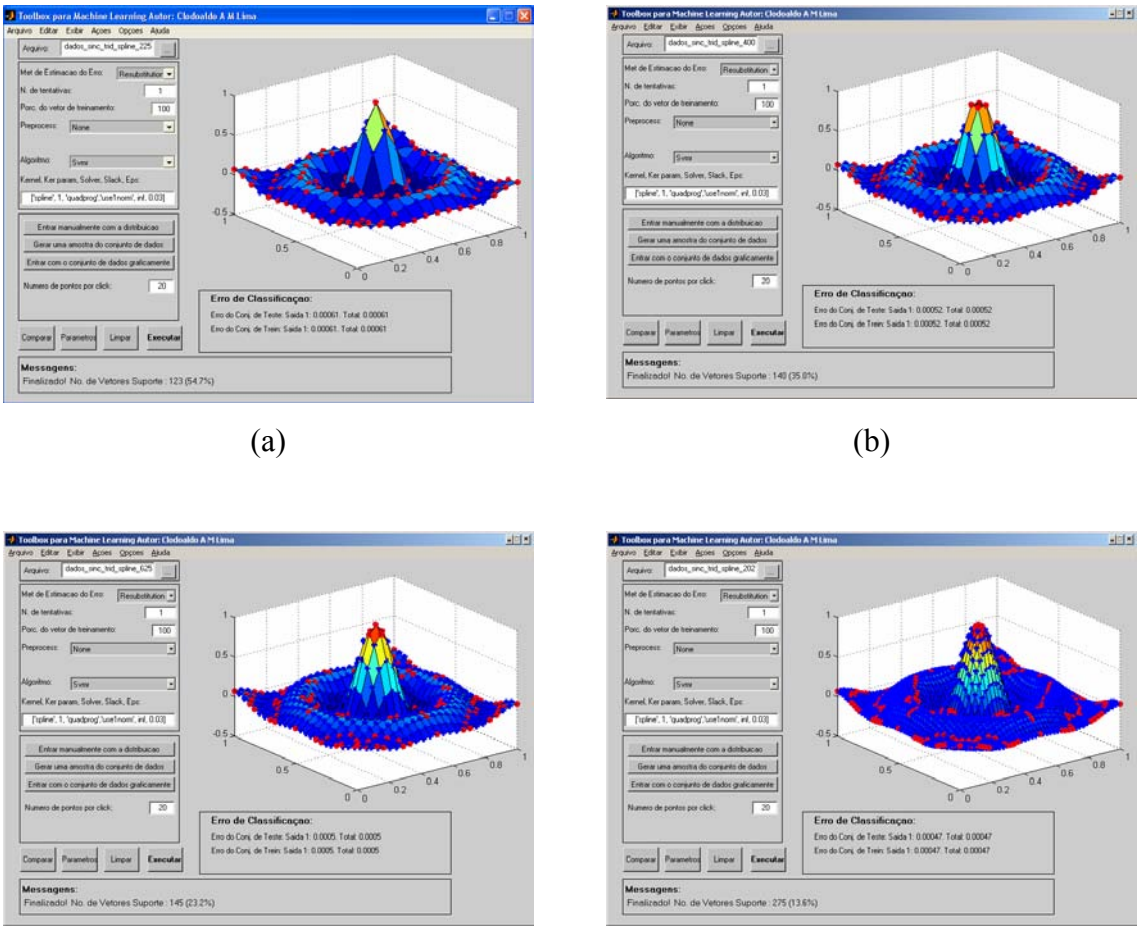


Figura 4.42 – Regressor SVM para função *sinc* bi-dimensional para a mesma precisão (a) 123 SVs para 225 amostras, (b) 140 SVs para 400 amostras, (c) 145 SVs para 625 amostras, (d) 275 SVs para 2025 amostras.

4.15 O controle do número de vetores-suporte pelo valor de ε

Suponha que se necessita aproximar uma função $f(\mathbf{x})$ com uma precisão ε, isto é, descrever a função $f(\mathbf{x})$ por uma outra função $f^*(\mathbf{x})$ tal que a função $f(\mathbf{x})$ esteja situada dentro de um 'tubo' de espessura ε, ou simplesmente tubo-ε, formado pela função $f^*(\mathbf{x})$ (veja a Figura 4.43). Para construir tal função, adota-se um tubo-ε elástico (um tubo que

tende a ser plano) e coloca-se a função $f(\mathbf{x})$ dentro deste tubo-ε. Uma vez que o tubo-ε tende a tornar-se plano, este irá 'tocar' em alguns pontos da função $f(\mathbf{x})$. Pode-se segurar o tubo exatamente nestes pontos. Então o eixo do tubo define a aproximação $f^*(\mathbf{x})$ da função $f(\mathbf{x})$ com uma precisão ε , e as coordenadas dos pontos onde o tubo ε toca a função $f(\mathbf{x})$ definem os vetores-suporte. O kernel $K(\cdot,\cdot)$ descreve a lei de elasticidade.

De fato, uma vez que a função $f(\mathbf{x})$ foi inserida no tubo-ε, não há pontos da função com distância maior que ε à linha central do tubo. Portanto, a linha central do tubo descreve a aproximação desejada.

Para ver quais pontos tocam o tubo-ε, os quais definem os vetores-suporte, basta lembrar que estes são obtidos solucionando um problema de otimização para o qual as condições de Karush-Kuhn-Tucker são válidas. De acordo com a definição de vetores-suporte, eles são aqueles para os quais os multiplicadores de Lagrange nas condições de Karush-Kuhn-Tucker são diferentes de zero. Estes multiplicadores definem os pontos da borda em um problema de otimização com restrições de desigualdade, isto é, coordenadas onde a função $f(\mathbf{x})$ toca o tubo-ε. Para um tubo-ε largo, haverá um número baixo de pontos tocando a função $f(\mathbf{x})$. Este modelo é válido para problemas de aproximação de funções com qualquer número de variáveis. A Figura 4.43 mostra o tubo de aproximação que corresponde ao caso de aproximação da função *sinc* unidimensional com precisão ε = 0,2.

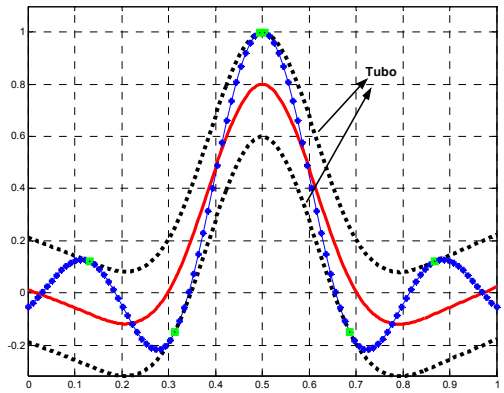


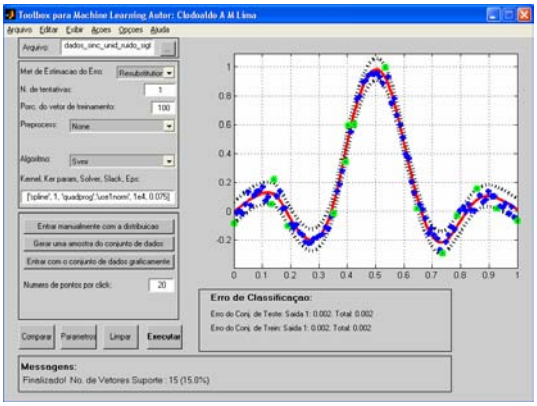
Figura 4.43 – Tubo-ε de aproximação referente à função *sinc* unidimensional.

4.16 Abordagem SVM para aproximação de funções com dados ruidosos

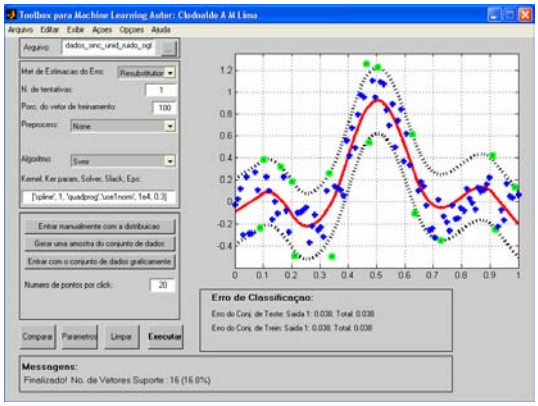
Nesta seção, será aplicada a abordagem SVM ao problema de aproximação da função *sinc* unidimensional, mas com a adição de ruído gaussiano com média zero e variância ζ. O objetivo desta seção é mostrar a relação empírica que existe entre a variância do ruído ζ e a precisão desejada ε.

A Figura 4.44 (a)-(d) apresenta os resultados dos experimentos de estimação a partir de dados corrompidos por diferentes níveis de ruído. Os círculos em verde na Figura 4.44 (a)-(d) indicam os vetores suporte. Na Figura 4.44 (a)-(c), foram realizados experimentos em que a precisão ε era sempre menor que a variância do ruído. Observe que, para este caso, o número de vetores suporte (SVs) não alterou significativamente, variando de 15 a 17.

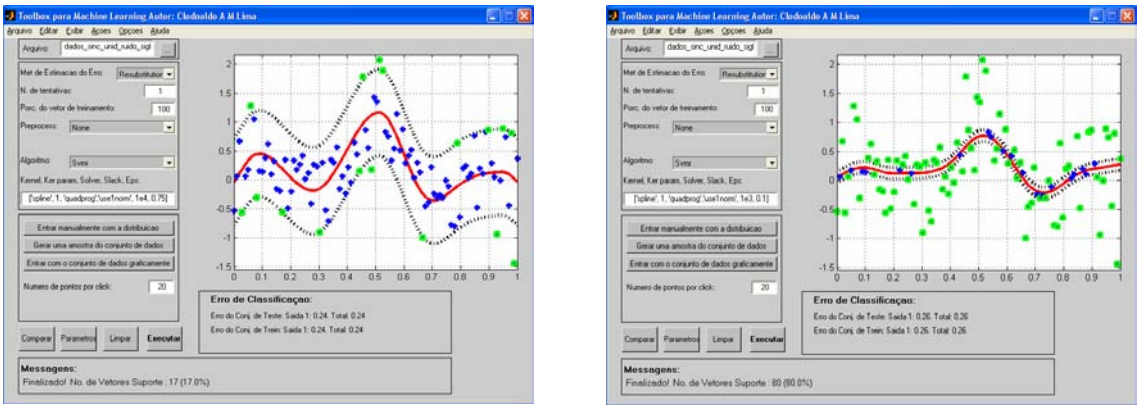
No entanto, quando a variância do ruído ζ é igual a 0,4 e a precisão é dada por $\varepsilon = 0{,}1$, ou seja, variância do ruído maior que a precisão ε, veja Figura 4.44 (d), o número de vetores suporte aumenta drasticamente. Resultados equivalentes podem ser obtidos usando a função *sinc* bidimensional. Portanto, pode-se concluir que a aplicação de SVM a dados ruidosos deve levar em consideração alguma técnica para estimação da variância do ruído. No capítulo 5, será apresentada uma extensão de SVM para ensembles que, dentre outras contribuições, tende a reduzir a influência do ruído no resultado global.



(a)



(b)



(c) (d)

Figura 4.44 – A função de regressão e sua aproximação obtida a partir dos dados com diferentes níveis de ruído e diferentes valores de ζ e ε; (a) ζ = 0,05, ε = 0,075 e 15 SVs; (b) ζ = 0,2, ε = 0,3 e 16 SVs; (c) ζ = 0,3, ε = 0,75 e 15 SVs; (d) ζ = 0,4, ε = 0,1 e 80 SVs.

4.17 Uma visão de SVM como uma rede neural

Nesta seção, será apresentado como uma SVM pode automaticamente definir a função de transferência e o número de neurônios em uma rede neural. A título ilustrativo, será utilizado o problema do OU-exclusivo, conforme descrito abaixo.

4.17.1 Problema do OU-exclusivo

Este exemplo demonstra a mecânica do cálculo do SVM para casos em que os dados são linearmente separáveis no espaço de características. O problema OU–exclusivo (XOR) pode ser definido como segue: encontre um hiperplano de separação que classifique sem erro o seguinte conjunto de dados:

Índice *i*	**x**	*y*
1	(1,1)	1
2	(1,-1)	-1
3	(-1,-1)	1
4	(-1,1)	-1

Tabela 4.4 – Conjunto de dados para o problema do OU - Exclusivo

No espaço original dos dados, não é possível solucionar este problema com uma superfície de decisão linear. Assim, uma superfície de decisão polinomial de ordem 2 será usada para separar estes dados. O produto interno de *kernel* polinômial de ordem 2 é dado por:

$$K(\mathbf{x},\mathbf{x}_i) = \left[(\mathbf{x}\cdot\mathbf{x}_i)+1\right]^2 . \tag{4.157}$$

Esta expressão corresponde a um conjunto de funções-base, $\varphi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1x_2, x_1^2, x_2^2)$, onde x_1, x_2 representam as duas coordenadas do espaço de entrada **x**. O vetor φ é um ponto no espaço de características de dimensão 6. Para determinar a superfície, deve-se maximizar o funcional a seguir, com $C \to \infty$:

$$W(\alpha) = \alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 - \frac{1}{2}\sum_{i,j}^{4} \alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j)$$

$$\text{sujeito a } \begin{cases} \sum_{i=1}^{4} y_i\alpha_i = \alpha_1 - \alpha_2 + \alpha_3 - \alpha_4 = 0 \\ \alpha_1 \geq 0,\ \alpha_2 \geq 0,\ \alpha_3 \geq 0,\ \alpha_4 \geq 0 \end{cases} \tag{4.158}$$

O produto interno do *kernel* é representado por uma matriz **K** 4×4 com elementos $K(\mathbf{x}_i,\mathbf{x}_j)$, calculados usando (4.157), produzindo:

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix} \tag{4.159}$$

A solução deste problema de otimização é $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0{,}125$, indicando que todos os quatro pontos do conjunto de treinamento são vetores-suporte. O funcional *W* encontra

um máximo de 0,25 para a solução ótima. A função de decisão na representação de produto interno é dada por:

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) = (0{,}125) \sum_{i=1}^{4} y[(\mathbf{x}_i \cdot \mathbf{x}) + 1]^2 \tag{4.160}$$

Esta função de decisão separa os dados com margem máxima. A margem pode ser calculada baseada no máximo de *W*, conduzindo a:

$$2W(\alpha) = \|\mathbf{w}\|^2 = 0{,}5\,, \tag{4.161}$$

$$\rho = \frac{1}{\|\mathbf{w}\|} = \sqrt{2}\,, \tag{4.162}$$

onde

$$\mathbf{w} = \sum_{i=1}^{4} \alpha_i y_i \varphi_i(\mathbf{x})\,. \tag{4.163}$$

Para este problema simples com um espaço de características de baixa dimensão, é possível escrever a função de decisão em termos de bases polinomiais:

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i \varphi(\mathbf{x}) = w_1 + w_2\sqrt{2}x_1 + w_3\sqrt{2}x_2 + w_4\sqrt{2}x_1x_2 + w_5x_1^2 + w_6x_2^2 \tag{4.164}$$

A equação (4.163) é usada para obter os parâmetros $w_1, w_2, w_3, w_4, w_5, w_6$ em termos de α_i, na forma:

$$\mathbf{w} = \sum_{i=1}^{4} \alpha_i y_i \varphi(\mathbf{x}_i) = [0 \quad 0 \quad 0 \quad \tfrac{1}{\sqrt{2}} \quad 0 \quad 0] \tag{4.165}$$

Resulta então a função de decisão (4.166), que pode ser vista como uma rede neural com um neurônio na camada intermediária (Figura 4.45):

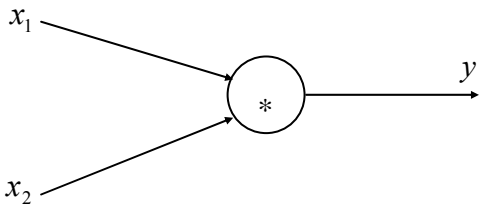


Figura 4.45 – Rede neural implementada pelo SVM.

$$f(x_1, x_2) = x_1 x_2 \,. \tag{4.166}$$

Analisando a saída da SVM para os dados de treinamento, observa-se que esta função de decisão está correta, como apresentado na Tabela 4.5. A Figura 4.46 apresenta o gráfico da função de decisão no espaço de entrada.

Índice *i*	Espaço de entrada **x**	Espaço de Características φ						Saída
	(x_1, x_2)	1	x_1	x_2	$x_1 x_2$	x_1^2	x_2^2	y
1	(1,1)	1	1	1	1	1	1	1
2	(1,-1)	1	1	-1	-1	1	1	-1
3	(-1,-1)	1	-1	-1	1	1	1	1
4	(-1,1)	1	-1	1	-1	1	1	-1

Tabela 4.5 – Classificação dos dados XOR.

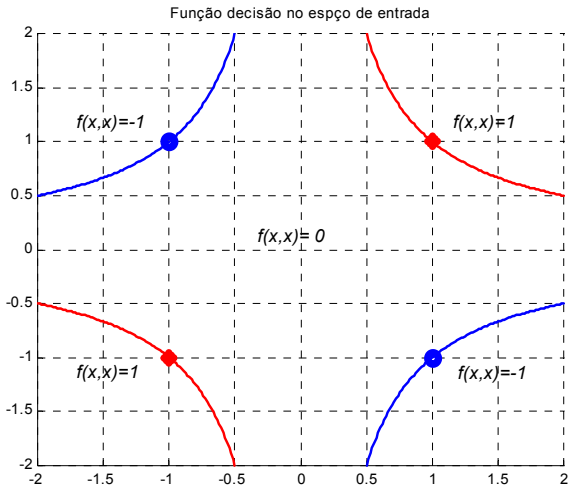


Figura 4.46 – Função de decisão no espaço de entrada.

Observe que:

- No espaço de entrada bi-dimensional, a função de decisão é não-linear;
- No espaço de características de dimensão 6, cujas coordenadas são $\varphi(\mathbf{x}) = (1, \sqrt{2}\, x_1, \sqrt{2}\, x_2, \sqrt{2}\, x_1 x_2, x_1^2, x_2^2)$, a função de decisão é linear e apresenta margem máxima.

4.18 Abordagens algorítmicas para treinamento de SVMs

O interesse crescente em aplicações de SVM para problemas de grande escala (problemas com um grande número de variáveis e amostras de treinamento) exige algoritmos de treinamento computacionalmente factíveis. Algumas aplicações recentes requerem o tratamento de 10^5 a 10^6 amostras (SMOLA & SCHÖLKOPF, 1998). Conforme já mostrado nas seções anteriores, o treinamento de uma SVM é obtido a partir da solução de um problema de programação quadrática (QP). O número de variáveis no problema de otimização é igual ao número de amostras do conjunto de treinamento (SVM para classificação de padrões) ou duas vezes o número de amostras do conjunto de treinamento (SVM para regressão). A velocidade dos algoritmos convencionais para solução de problemas quadráticos de propósito geral é insuficiente para problemas contendo algumas centenas de amostras. Sendo assim, a busca por algoritmos de treinamento de propósito específico tem levado a várias implementações alternativas, que procuram tirar vantagem da estrutura particular do problema de treinamento de uma SVM.

Um pacote disponível comercialmente para QP é o OSL (*Optimization Subroutine Library*) produzido pela IBM. Ele usa um algoritmo de duas fases para minimizar uma função quadrática com a matriz quadrática semidefinida positiva sujeita a restrições lineares. Uma vez que o ótimo pode ocorrer no interior da região de factibilidade, o método simplex não pode ser usado para solucionar problemas QP. O primeiro módulo do algoritmo soluciona um problema de programação linear aproximado, usando o método simplex, e associa um problema QP muito simples a cada iteração. Quando as aproximações sucessivas são bastante próximas (variação na função-objetivo é pequena), é usado um segundo módulo do algoritmo, que admite uma função-objetivo quadrática e converge rapidamente a partir de um bom valor de partida.

Outro pacote, MINOS do Stanford Optimization Laboratory (MURTAGH & SAUNDERS, 1983) usa um algoritmo gradiente reduzido junto com um algoritmo quase-Newton. As restrições são controladas por uma estratégia de conjunto ativo. A factibilidade é mantida em todo o processo. As variáveis são classificadas como básicas, super-básicas e não-básicas; na solução, as variáveis básicas e super-básicas são aquelas que estão distantes

de seus limitantes. O espaço nulo da matriz de kernel é construído a partir dos coeficientes da matriz de variáveis básicas, usando fatoração esparsa. Considerando as diversas restrições ativas, uma aproximação quase-Newton para a matriz de Kernel (Hessiana) reduzida é realizada.

Finalmente, um sistema proposto por Kaufmann (DRUCKER *et al*., 1997), usa um método chamado conjunto *iterativo livre*, o qual inicializa com todas as variáveis no seu limitante inferior e adiciona aquelas que violam a condição Karush-Kuhn-Tucker. Esta abordagem tem a grande vantagem de não ter que computar todo o produto interno no início. Ao invés disso, é avaliado durante a execução, produzindo uma melhoria de desempenho comparável às estratégias que tratam todo o problema de otimização de uma vez.

Já quanto ao uso do toolbox de otimização do MATLAB, embora imediato, este não é recomendado para mais de 100 amostras de treinamento. Isto se deve ao fato de que o método utilizado não é eficiente para tratar problemas de otimização de tamanho elevado e onde pelo menos metade dos autovalores da matriz de *kernel* torna-se muito pequeno. Um aumento considerável de velocidade pode ser obtido adaptando a estratégia de otimização para a situação particular da abordagem SVM.

Portanto, MINOS, e MATLAB são alguns dos pacotes QP prontos que são aplicáveis ao treinamento do SVM, com certas restrições. Estes métodos podem ser usados para treinar uma SVM rapidamente, mas eles apresentam a desvantagens de requerer que a matriz de *kernel* seja armazenada na memória. Para conjuntos de dados pequenos, este procedimento é prático e estas rotinas QP são a melhor escolha. Mas para conjuntos de dados grandes, técnicas alternativas devem ser adotadas.

Há na literatura duas categorias de técnicas alternativas que visam tratar o problema do armazenamento da matriz de *kernel*: técnicas nas quais os componentes do *kernel* são avaliados e descartados durante o aprendizado e aquelas na qual o conjunto de trabalho é um subconjunto dos dados de treinamento utilizado. Para a primeira categoria, a abordagem mais óbvia é atualizar seqüencialmente os coeficientes α_i e esta é uma abordagem usada pelo algoritmo *Kernel Adatron* (FRIESS *et al*., 1998). Para classificação binária (onde o problema é linearmente separável no espaço de característica e sem o termo de bias),

resulta em um procedimento simples, usando gradiente ascendente. Há uma equivalência com o método Hildreth na teoria de otimização, o qual pode ser generalizado para o caso de margens suaves e inclusão de um bias (LUENBERGER, 1984). Entretanto, o algoritmo resultante pode não ser tão rápido quanto muitas rotinas QP, especialmente para conjuntos de dados pequenos.

Ao contrário da atualização seqüencial dos coeficientes α_i, a alternativa é atualizar os coeficientes α_i em paralelo, mas usando somente um subconjunto (ou *chunk*) dos dados para cada estágio. Assim, uma rotina QP é usada para otimizar o lagrangeano sobre um subconjunto arbitrário de dados. Os vetores-suporte encontrados são guardados e todas as demais amostras (com $\alpha_i = 0$) são descartadas. Um novo conjunto de trabalho é então derivado a partir destes vetores-suporte e as amostras adicionas que mais violam as restrições são armazenadas. Este processo de *chunking* é então repetido até que a margem seja maximizada. É evidente que este procedimento pode falhar, quando o conjunto de dados é muito grande ou a hipótese de modelagem dos dados não é esparsa (muitos dos α_i não são zero). Neste caso, métodos de decomposição (OSUNA *et al.*, 1999) fornecem uma melhor abordagem. Este algoritmo somente usa um subconjunto fixo dos dados com os α_i restantes mantido fixos.

A principal direção de pesquisa em algoritmos de treinamento para SVM é o método de decomposição. A idéia-chave da decomposição, devida a OSUNA *et al.* (1997), é congelar todas as variáveis, com exceção de um número pequeno de variáveis de otimização, e solucionar uma seqüência de pequenos problemas de tamanho fixo. O conjunto de variáveis cujos valores são otimizados na iteração atual é chamado *conjunto de trabalho*. A complexidade de re-otimização do conjunto de trabalho é suposta constante em termos de tempo. A fim de que um algoritmo de decomposição seja bem sucedido, o conjunto de trabalho deve ser selecionado de forma criteriosa. Um dos algoritmos de decomposição mais rápidos foi proposto por JOACHIMS (1998). Ele é baseado no método Zoutendijk's de direções factíveis, proposto junto à comunidade de otimização no início de 1960 para selecionar um bom conjunto de trabalho. Entretanto, o algoritmo de JOACHIMS (1998) é limitado a SVMs aplicadas à classificação de padrões, pois este faz uso de rótulos

±1. A estratégia de *shrinking*[2] e a diretriz de *caching*[3] foram também usadas para aumentar a velocidade de treinamento. Uma falha na estratégia *shrinking* resultará na re-execução do problema de aprendizado. LASKOV (2000) apresentou um algoritmo similar para regressão.

O caso limite da decomposição é o algoritmo de otimização seqüencial mínima (SMO) de PLATT (1999) na qual somente dois coeficientes α_i são otimizados a cada iteração. O menor conjunto de parâmetros que pode ser otimizado em cada iteração é claramente dois se a restrição $\sum_{i=1}^{N} \alpha_i y_i = 0$ é válida. Pode-se observar que, se somente dois parâmetros são otimizados e o restante é mantido fixo, então é possível derivar uma solução analítica que pode ser executada usando poucas operações numéricas. O método, portanto, consiste de um passo heurístico para encontrar o melhor par de parâmetros a serem otimizados e usa uma expressão analítica para assegurar o incremento monotônico do lagrangeano. O algoritmo SMO foi refinado para melhorar a velocidade (KEERTHI *et al.*, 1999) e generalizado para tratar problemas de classificação (PLATT, 1999), regressão (SMOLA & SCHÖLKOPF, 1998) e estimação de densidades (SCHÖLKOPF *et al.*, 1999). Devido a esta decomposição da tarefa de aprendizado e levando em conta a velocidade, estas propostas de algoritmos são as mais indicadas para treinamento de SVMs.

Algumas heurísticas são sugeridas para selecionar o conjunto de trabalho. KEERTHI *et al*. (2001) adicionou melhoria de desempenho ao SMO, apontando a ineficiência na atualização do parâmetro de bias no algoritmo de Platt e o substituiu por dois parâmetros. Além disso, é importante observar que KEERTHI & GILBERT (2002) provaram que a condição de parada se estabelece em um número finito de passos.

DECOSTE & SHÖLKOPF (2002) mostraram que o número de candidatos a vetores-suporte durante os primeiros estágios de treinamento é muito maior que o número de vetores-suporte no estágio final. Muitos resultados experimentais mostram que a complexidade temporal do SMO pode ser aproximada para O(*L.N*), onde *N* é o tamanho do conjunto de treinamento e *L* é o número médio de vetores-suporte durante as iterações. Uma redução eficiente de *L* terá um impacto importante sobre o desempenho do SMO.

[2] A idéia de *shrinking* é remover algumas variáveis cujos valores permanecem iguais a zero ou *C* por um longo tempo, e que provavelmente não mudarão.
[3]Algumas linhas da matriz hessiana são guardadas na memória, ao invés da matriz completa.

Portanto, DECOSTE & SHÖLKOPF (2002) introduziram a idéia de *digest* para explorar esta propriedade. Quando o número de vetores-suporte passa de um limite superior, a idéia é comutar o processo iterativo utilizando SMO para um processo no qual a iteração dos candidatos a vetores-suporte não tem limitante superior. Esta estratégia visa condensar este conjunto de vetores-suporte candidatos, reduzindo o número de reavaliações do *kernel*. Entretanto, a heurística de DECOSTE & SHÖLKOPF (2002) contém muitos parâmetros arbitrários e as estratégias de cache das linhas da matriz de *kernel* é ainda ineficiente. FLAKE & LAWRENCE (2002) mostraram que o SMO acessa a matriz de *kernel* de forma irregular. Uma diretriz simples de caching como LRU (*Least Recently Used*) pode resultar em uma degradação de desempenho do SMO.

Em DONG *et al.* (2003), uma integração efetiva da idéia de decomposição, *caching*, *digest* e *shrinking* é aplicada ao algoritmo SMO aperfeiçoado por KEERTCH *et al*. (2001) para encontrar desempenho promissor, onde o *caching* da matriz de *kernel* possui um papel mais importante. A dimensão da matriz quadrada de cache do *kernel* é a mesma do tamanho do conjunto de trabalho, que não é menor que o número de vetores-suporte finais. *Digest* restringe o crescimento dos candidatos a vetores-suporte tal que o número de candidatos do conjunto de vetores-suporte durante o treinamento é sempre menor que o tamanho do conjunto de trabalho. Além disso, maximiza-se o re-uso da matriz de cache do *kernel*, que é um passo importante. São apresentadas também sugestões para a seleção do conjunto de trabalho e para a regra de parada do método de decomposição.

Nesta tese, optou-se por utilizar a abordagem proposta por JOACHIMS (1998), tanto para problemas de regressão quanto para problemas de classificação. Como esta abordagem foi proposta inicialmente para classificação, ela é estendida nesta tese para o tratamento de problemas de regressão, considerando vários tipos de funções de perda apresentadas em seções anteriores.

4.19 Máquina de Vetores-Suporte baseada em Quadrados Mínimos

Uma das variações de SVM propostas na literatura que tem alcançado bastante sucesso em problemas de classificação e regressão baseia-se no critério dos quadrados mínimos (SUYKENS & VANDERWALLE, 1999). Nesta formulação, utilizam-se restrições de igualdade ao invés de restrições de desigualdade e a função-custo é a soma do erro quadrático (SSE – *Squared Sum of Error*), como é freqüentemente usada no treinamento de redes neurais. Esta reformulação simplifica muito o problema, de forma que a solução é caracterizada por um sistema linear, mais precisamente um sistema KKT (Karush-Khun-Tucker) (FLETCHER, 1987). Este sistema é muito similar a um sistema de equações que é solucionado a cada iteração pelo método dos pontos interiores para SVM (SMOLA, 1999). Este sistema linear pode ser eficientemente solucionado por métodos iterativos tais como gradiente conjugado (SUYKENS *et al.*, 1999). Enquanto o SVM padrão foi mais intensamente aplicado a problemas estáticos, o fato do LS-SVM usar restrição de igualdade e uma formulação baseada no SSE permite estendê-lo para o tratamento de estruturas de processamento que apresentam dinâmica, como por exemplo redes recorrentes (SUYKENS & VANDERWALLE, 2000) e para aplicações em controle (SUYKENS *et al.*, 2002).

Entretanto, apesar das características atrativas computacionalmente, a solução baseada em LS-SVM também tem algumas limitações. A primeira limitação está associada à perda da natureza esparsa do problema, pois todas as amostras do conjunto de treinamento estão contribuindo para o modelo e a importância relativa delas é dada pelo seu valor suporte. Em outras palavras, todas as amostras passam a desempenhar o papel de vetores-suporte. Uma segunda limitação é que o uso da função custo SSE sem regularização pode conduzir a estimativas que são menos robustas, principalmente quando há *outliers* nos dados de treinamento ou quando não se pode supor que o erro cometido pelo modelo tenha uma distribuição gaussiana.

Para atenuar tais limitações, SUYKENS *et al.* (2002) propuseram uma versão ponderada do LS-SVM. Nesta versão, introduz-se uma variável que pondera a importância dos erros cometidos pelo modelo na função custo. Na metodologia padrão de SVM,

escolhe-se uma dada função custo (qualquer função custo convexa pode ser adotada, em princípio, como mostrado em SMOLA (1999)). A abordagem LS-SVM ponderada implementa uma seqüência de LS-SVMs ponderadas a partir de uma versão não-ponderada. Desta forma, tenta-se implicitamente encontrar uma função-custo ótima, ao invés de impor a função custo antecipadamente. Neste sentido, há também um elo entre a solução do problema de treinamento de uma SVM, para uma função custo convexa, pelo método de pontos interiores e uma solução via LS-SVM com ponderação iterativa. Enquanto na abordagem proposta por SUYKENS *et al.* (2002) a ponderação da função-custo é encontrada iterativamente via um processo heurístico, LIMA *et al.* (2004) apresentou uma formulação de LS-SVM para mistura de especialistas e mostrou que tal ponderação emerge naturalmente do processo de treinamento dos especialistas.

Nas próximas seções, será apresentado o formalismo de LS-SVM para problemas de classificação e regressão.

4.19.1 LS-SVM para problemas de classificação

Dado um conjunto de treinamento composto por N amostras $\{\mathbf{x}_k, y_k\}_{k=1}^{N}$, onde $\mathbf{x}_k \in \Re^n$ é o k-ésimo padrão de saída e $y_k \in \{\pm 1\}$, a abordagem LS-SVM objetiva a construção de um classificador da seguinte forma:

$$f(\mathbf{x}) = sign[\sum_{k=1}^{N} \alpha_k y_k K(\mathbf{x}, \mathbf{x}_k) + b], \tag{4.167}$$

onde α_k são constantes reais positivas e b é uma constante real.

Aqui, se introduz uma versão dos quadrados mínimos para classificador SVM formulando o problema de classificação como:

$$J(\mathbf{w}, b, e) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{2}\sum_{k=1}^{N} e_k^2, \tag{4.168}$$

sujeito às restrições de igualdade:

$$y_k[\mathbf{w}^T\varphi(\mathbf{x}_k) + b] = 1 - e_k, \quad k = 1,..., N. \tag{4.169}$$

Define-se o Lagrangeano como:

$$L(w,b,e,\alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{2}\sum_{k=1}^{N} e_k^2 - \sum_{k=1}^{N}\alpha_k\{y_k[\mathbf{w}^T\varphi(\mathbf{x}_k)+b]-1+e_k\}, \qquad (4.170)$$

onde α_k são multiplicadores de Lagrange (que podem ser positivos ou negativos agora, devido às restrições de igualdade, como segue da condição de Kuhn-Tucker-Karush (FLETCHER, 1987)).

As condições para otimalidade são:

$$\begin{cases} \dfrac{\partial L}{\partial w} = 0 \rightarrow \mathbf{w} = \sum_{k=1}^{N}\alpha_k y_k \varphi(\mathbf{x}_k) \\ \dfrac{\partial L}{\partial b} = 0 \rightarrow \sum_{k=1}^{N}\alpha_k y_k = 0 \\ \dfrac{\partial L}{\partial e_k} = 0 \rightarrow \alpha_k = Ce_k \\ \dfrac{\partial L}{\partial \alpha_k} = 0 \rightarrow y_k[\mathbf{w}^T\varphi(\mathbf{x}_k)+b]-1+e_k = 0,\ k = 1,...,N \end{cases} \qquad (4.171)$$

As equações podem ser escritas imediatamente como um conjunto de equações lineares (FLETCHER, 1987), produzindo:

$$\begin{bmatrix} \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 1 & -\mathbf{Y}^T \\ 0 & 0 & C\mathbf{I} & -\mathbf{I} \\ \mathbf{Z} & \mathbf{Y} & \mathbf{I} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{w} \\ b \\ \mathbf{e} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vec{1} \end{bmatrix} \qquad (4.172)$$

onde $\mathbf{Z} = [\varphi(\mathbf{x}_1)^T y_1;...;\varphi(\mathbf{x}_N)^T y_N]$, $\mathbf{Y} = [y_1,...y_N]$, $\vec{\mathbf{1}} = [1;...;1]$, $\mathbf{e} = [e_1;...;e_N]$ e $\boldsymbol{\alpha} = [\alpha_1;...;\alpha_N]$.

Depois da eliminação de **w** na equação (4.172), obtém-se a solução pode ser obtida por:

$$\begin{bmatrix} 0 & -\mathbf{Y}^T \\ \mathbf{Y} & \mathbf{Z}\cdot\mathbf{Z}^T + C^{-1}\mathbf{I} \end{bmatrix}\begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{\mathbf{1}} \end{bmatrix} \qquad (4.173)$$

A condição de Mercer pode ser aplicada novamente para a matriz $\boldsymbol{\Omega} = \mathbf{Z}\cdot\mathbf{Z}^T$, onde:

$$\Omega(\mathbf{x}_k,\mathbf{x}_l) = y_k y_l \varphi(\mathbf{x}_k)^T\varphi(\mathbf{x}_l), \qquad (4.174)$$

ou seja,

$$\Omega(\mathbf{x}_k,\mathbf{x}_l) = y_k y_l K(\mathbf{x}_k,\mathbf{x}_l). \qquad (4.175)$$

Assim, o classificador (4.167) pode ser obtido solucionando o conjunto de equações lineares (4.173-4.175), ao invés de recorrer à programação quadrática. Os valores-suporte são proporcionais aos erros no conjunto de treinamento (veja restrições 4.171), enquanto no caso do SVM tradicional muitos valores são iguais a zero. Assim, pode-se dizer que há um espectro de valores suporte no caso dos quadrados mínimos.

4.19.2 LS-SVM para problemas de regressão

Dado um conjunto de *N* amostras de treinamento $\{\mathbf{x}_k, y_k\}_{k=1}^{N}$ com entradas $\mathbf{x}_k \in \Re^n$ e saída $y_k \in \Re$, considera-se o seguinte problema de otimização:

$$\min_{\mathrm{w},b,e} J(\mathbf{w},b,e) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \frac{C}{2}\sum_{k=1}^{N} e_k^2 \quad , \tag{4.176}$$

tal que

$$y_k = \mathbf{w}^T\varphi(\mathbf{x}_k) + b + e_k \quad k = 1,...,N \quad , \tag{4.177}$$

onde $\varphi(\cdot): \Re^n \to \Re^{n_h}$ é uma função que mapeia o espaço de entrada em um espaço de alta dimensionalidade, chamado de espaço de características, $\mathbf{w} \in \Re^{n_h}$ é o vetor de pesos na formulação primal, $e_k \in \Re$ é a variável erro e *b* é o termo de bias. Observe que a função custo *J* consiste da soma do erro quadrático de ajuste e um termo de regularização, relacionado com regressão *ridge*[4] (GOLUB & VAN LOAN, 1989). A importância relativa destes termos é determinada por uma constante real *C* (alguns outros autores utilizam o símbolo γ). No caso de dados ruidosos, pode-se evitar sobre-ajustes adotando um valor pequeno para *C*. Esta formulação do problema LS-SVM foi independemente investigado por SAUNDERS *et al.* (1998) (sem o termo bias) e SUYKENS & VANDEWALLE (1999).

Na formulação primal, tem-se o modelo:

$$f(\mathbf{x}) = \mathbf{w}^T\varphi(\mathbf{x}) + b \, . \tag{4.178}$$

Define-se o Lagrangeano, como:

[4] *Ridge Regression* é um tipo de regularização.

$$L(\mathbf{w},b,e,\alpha)=\frac{1}{2}\mathbf{w}^T\mathbf{w}+\frac{C}{2}\sum_{k=1}^{N}e_k^2-\sum_{k=1}^{N}\alpha_k\{\mathbf{w}^T\varphi(\mathbf{x}_k)+b+e_k-y_k\}, \tag{4.179}$$

com multiplicadores $\alpha_k \in \Re$ (chamados de valores-suporte). As condições de otimalidade são dadas por:

$$\begin{cases}\dfrac{\partial L}{\partial w}=0\rightarrow \mathbf{w}=\sum_{k=1}^{N}\alpha_k\varphi(\mathbf{x}_k)\\ \dfrac{\partial L}{\partial b}=0\rightarrow\sum_{k=1}^{N}\alpha_k=0\\ \dfrac{\partial L}{\partial e_k}=0\rightarrow\alpha_k=Ce_k,k=1,...,N\\ \dfrac{\partial L}{\partial \alpha_k}=0\rightarrow \mathbf{w}^T\varphi(\mathbf{x}_k)+b+e_k-y_k=0,k=1,...,N\end{cases} \tag{4.180}$$

Estas condições são similares às condições de otimalidade SVM, exceto para a condição $\alpha_k = Ce_k$. É neste ponto que se perde a condição de se operar com um número reduzido de vetores-suporte (GIROSI, 1998)

Depois da eliminação de **w**, obtém-se a solução:

$$\begin{bmatrix}0 & 1_v^T\\ 1_v & \boldsymbol{\Omega}+C^{-1}\mathbf{I}\end{bmatrix}\begin{bmatrix}b\\ \boldsymbol{\alpha}\end{bmatrix}=\begin{bmatrix}0\\ \mathbf{Y}\end{bmatrix}, \tag{4.181}$$

com $\mathbf{Y}=[y_1,...,y_N]$, $\mathbf{1}_v=[1,...,1]$, $\boldsymbol{\alpha}=[\alpha_1;...;\alpha_N]$, e $\Omega(\mathbf{x}_k,\mathbf{x}_l)=\varphi(\mathbf{x}_k)^T\varphi(\mathbf{x}_1)$, para $k,l=1,...,N$.

Utilizando a condição de Mercer, o modelo LS-SVM resultante para regressão torna-se:

$$f(\mathbf{x})=\sum_{k=1}^{N}\alpha_k K(\mathbf{x},\mathbf{x}_k)+b\,. \tag{4.182}$$

onde α e b são solução da equação matricial (4.181).

Quando solucionando sistemas lineares grandes, é necessário aplicar métodos iterativos (GOLUB & LOAN, 1989) para a equação matricial (4.181). Entretanto, a matriz em (4.181) não é definida positiva. De acordo com (SUYKENS *et al.*, 1999), pode-se positivar a

matriz de modo a permitir a aplicação de vários métodos iterativos (gradiente conjugado, relaxação sucessiva e outros). Observe que a complexidade computacional do método do gradiente conjugado para solução do sistema linear $\mathbf{Ax} = \mathbf{B}$ é $O(Nr^2)$ onde posto($\mathbf{D}$) = r, com $\mathbf{A} = \mathbf{I} + \mathbf{D}$ e $\mathbf{A} \in \Re^{N \times N}$. Ele converge em r+1 passos. A velocidade de convergência depende da escolha do parâmetro C e do parâmetro do *kernel*.

4.19.3 Estimação robusta através de LS-SVM ponderado

A fim de obter uma estimação robusta baseada na solução LS-SVM apresentada acima, pode-se ponderar a variável erro $e_k = \alpha_k / C$ pelos fatores de ponderação v_k. Resulta o seguinte problema de otimização:

$$\min_{\mathbf{w}^*, b^*, e^*} J(\mathbf{w}, e) = \frac{1}{2}\mathbf{w}^{*T}\mathbf{w}^* + \frac{C}{2}\sum_{k=1}^{N} v_k e_k^{*2} \tag{4.183}$$

tal que

$$y_k = \mathbf{w}^{*T}\varphi(\mathbf{x}_k) + b^* + e_k^*, \;\; k = 1,..., N \, .$$

O lagrangeano torna-se:

$$l(\mathbf{w}, b, e, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\frac{1}{2}\sum_{k=1}^{N} v_k e_k^2 - \sum_{k=1}^{N} \alpha_k \{\mathbf{w}^T\varphi(\mathbf{x}_k) + b + e_k - y_k\} \, . \tag{4.184}$$

As variáveis desconhecidas para este problema LS-SVM ponderado são denotadas pelo símbolo *. A partir das condições para otimalidade e eliminação de **w**, obtém-se o sistema KKT:

$$\begin{bmatrix} 0 & 1_v^T \\ \mathbf{1}_v^T & \mathbf{\Omega} + \mathbf{V}_c \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{Y} \end{bmatrix}, \tag{4.185}$$

onde a matriz diagonal é dada por:

$$\mathbf{V}_c = diag\left\{\frac{1}{Cv_1}, ..., \frac{1}{Cv_N}\right\}. \tag{4.186}$$

A escolha dos pesos v_k é baseada na variável de erro $e_k = \alpha_k / C$ do LS-SVM da equação (4.181) (caso não-ponderado). Estimações robustas para v_k podem ser obtidas em ROUSSEEUW & LEROY (1987). Então, pode-se adotar:

$$v_k = \begin{cases} 1 & se \quad \left| e_k / \hat{s} \right| \le c_1 \\ \dfrac{c_2 - \left| e_k / \hat{s} \right|}{c_2 - c_1} & se \quad c_2 \le \left| e_k / \hat{s} \right| \le c_2 \\ 10^{-4} & caso \ \ contrário \end{cases} \tag{4.187}$$

onde $\hat{s}$ é uma estimativa robusta do desvio padrão da variável erro e_k do LS-SVM, como segue:

$$\hat{s} = \frac{IQR}{2 \times 0{,}6745} \tag{4.188}$$

onde o intervalo interquartil (IQR) é a diferença entre o 75º percentil e o 25º percentil.

As restrições são tipicamente escolhidas como c_1=2,5 e c_2=3 (ROUSSEEUW & LEROY, 1987). Esta é uma escolha razoável, levando-se em conta o fato de que, para uma distribuição gaussiana haverá poucos resíduos maiores que $2{,}5\,\hat{s}$. Outra possibilidade é determinar c_1 e c_2 a partir da estimação de densidade da distribuição e_k.

Converge-se então para o seguinte algoritmo:

- Solucione o sistema de equações descrito em (4.181);
- Calcule $e_k = \alpha_k / C$;
- Compute $\hat{s}$ a partir da distribuição e_k;
- Determine os pesos v_k baseado em e_k, $\hat{s}$;
- Solucione o LS-SVM ponderado (4.185), gerando o modelo $f(\mathbf{x}) = \sum_{k=1}^{N} \alpha_k K(\mathbf{x}, \mathbf{x}_k) + b$.

4.20 Seleção de parâmetros

Sabe-se que o desempenho de generalização de uma SVM depende de uma boa escolha dos parâmetros C e ε e dos parâmetros do *kernel*. Com a finalidade de aplicar a abordagem SVM para problemas práticos, a escolha adequada de tais parâmetros é essencial. O problema de seleção dos parâmetros ótimos é bastante complexo pelo fato de que a complexidade do modelo SVM (portanto, o desempenho de generalização) depende de todos os três tipos de parâmetro simultaneamente. As implementações de software existentes para regressão e classificação usualmente tratam os parâmetros da SVM como entradas definidas pelo usuário. Esta seção, focaliza atenção na escolha dos parâmetros C e ε. A seleção de um tipo particular de *kernel* e os parâmetros da função *kernel* são usualmente baseadas no conhecimento do usuário sobre o domínio da aplicação e podem refletir a dispersão dos valores de entrada $\mathbf{x}$ do conjunto de dados de treinamento (CHAPELLE & VAPNIK, 1999; SCHOLKOPF *et al.*, 1999; VAPNIK, 1998; 1999). Por exemplo, CHERKASSKY & MA (2004) mostraram exemplos de regressão com SVM usando função *kernel* de base radial (RBF), onde o parâmetro de dispersão (variância) da RBF refletia a dispersão dos valores de entrada $\mathbf{x}$ do conjunto de treinamento. No Capítulo 5, será mostrado que é possível evitar a escolha arbitraria do *kernel* e de seus parâmetros utilizando ensembles.

Existem abordagens práticas para escolha de C e ε que podem ser resumidas como segue:

- Parâmetros C e ε são selecionados pelo usuário baseado no conhecimento a priori (VAPNIK, 1998; 1999). Obviamente, esta abordagem não é apropriada para usuários que não são especialistas na abordagem SVM. Baseado na observação que o SVs residem fora do tubo-ε e a complexidade do modelo SVM depende do número de SVs, SCHOLKOPF *et al.* (1998) sugeriu que um outro parâmetro v, associado à fração de amostras fora do tubo-ε deveria ser controlado, ao invés de ε. Nesta abordagem, o parâmetro v tem que ser definido pelo usuário. Similarmente, MATTERA & HAYKIN (1999) propuseram escolher o valor de ε assim que a porcentagem de SVs no modelo de regressão seja por volta de 50% do número de amostras. Entretanto,

pode-se facilmente apresentar exemplos em que o desempenho de generalização ótimo é encontrado com um número de SVs maior ou menor que 50%.

- KWOK (2001) e SMOLA *et al.* (1998) propuseram valores de ε assintoticamente ótimos que são proporcionais à variância do ruído (CHERKASSKY & MULIER, 1998, VAPNIK, 1998; 1999). O principal problema prático de tal proposta é que não reflete o tamanho da amostra. Intuitivamente, o valor de ε deveria ser menor para conjuntos de amostras grandes (quando os dados têm o mesmo nível de ruído).
- MATTERA & HAYKIN (1999) selecionaram o parâmetro *C* igual ao intervalo de excursão dos valores da saída. Esta é uma proposta razoável, mas não leva em conta possíveis efeitos de *outliers* nos dados de treinamento.
- CHERKASSKY & MULIER (1998) e SCHOLKOPF *et al.* (2001) propuseram validação cruzada para seleção dos parâmetros. No entanto, esta é muita custosa computacionalmente.
- Alguns pesquisadores têm recentemente apresentado uma interpretação estatística de SVM para regressão (SMOLA & SCHOLKOPF, 1998; HASTIE *et al.*, 2001), onde a função de perda usada para o risco empírico (4.2) é relacionada ao tipo particular de ruído aditivo na formulação de regressão, isto é, $y = f(\mathbf{x}) + \zeta$, onde ζ é um erro (ruído) de média zero distribuído identicamente e independentemente. Dentro desta abordagem, o valor do parâmetro ε pode ser otimamente sintonizado para uma densidade de ruído particular, ao passo que o parâmetro C é interpretado como um parâmetro de regularização tradicional na formulação (4.52) e (4.53), que é usualmente estimado via validação cruzada (HASTIE *et al.*, 2001).
- CHERKASSKY & MA (2004) propuseram seleção analítica do parâmetro C diretamente dos dados de treinamento (sem recorrer a re-amostragem) e seleção analítica do parâmetro ε baseado no nível de ruído nos dados de treinamento (conhecido ou estimado) e no número de exemplos de treinamento.

Do exposto acima, há varias propostas (até mesmo conflitantes) para a escolha ótima dos parâmetros de regressão para SVM. Um estudo teórico e prático sobre a viabilidade de tais abordagens deve ser realizado para se obter consenso a respeito da

escolha adequada de tais parâmetros. Nesta tese, apesar de ser custoso computacionamente, adotar-se-á a validação cruzada como critério de seleção de parâmetros, tanto para classificação quanto para regressão. Na seção seguinte, apresenta-se o resultado obtido para um problema de bioinformática, usando validação cruzada para selecionar o melhor parâmetro *C* e o parâmetro do *kernel* dentro de um intervalo especificado a priori.

4.20.1 Diagnostico de câncer de cólon

Em ALON *et al*. (1999), os autores descreveram e estudaram um conjunto de dados constituído de 62 amostras de tecido, cada uma com 2000 atributos (níveis de expressão gênica). Dentre os 62 tecidos, 22 são considerados normais e 40 apresentam câncer. No Apêndice A, será apresentada uma descrição detalhada deste conjunto.

Com o intuito de mostrar que o desempenho da abordagem SVM depende dos parâmetros escolhidos, tanto do parâmetro do *kernel* quanto do parâmetro *C*, foi adotado um intervalo de busca para cada parâmetro e utilizado um método de validação cruzada com 10 partições como critério de medida de desempenho do modelo, para cada par de parâmetros escolhido.

A busca adotada foi realizada em dois níveis. No primeiro nível, foi gerada uma matriz 10×10 contendo os valores dos parâmetros no intervalo especificado. Para cada par de parâmetros, foi utilizada validação cruzada com 10 partições (Figura 4.47). No segundo nível, para os melhores valores dos parâmetros obtidos no primeiro nível, foi gerada uma nova matriz 10×10 contendo os valores dos parâmetros em torno destes melhores valores. Novamente, para cada um dos parâmetros, foi utilizada validação cruzada com 10 partições (Figura 4.48). Os resultados para os diferentes tipos de *kernel* empregados, com os respectivos intervalos de busca, são apresentados na Tabela 4.6. Na ultima coluna da Tabela 4.6 é apresentado o desempenho de cada tipo de *kernel*, observe que os valores obtidos, para este problema, foram todos iguais. Isto deve-se principalmente a natureza do problema, não podendo ser generalizado para outros casos.

Tipo de Kernel	Intervalo de Busca		Valores Ótimos		Melhor Resultado
	C	Param	*C*	Param	
Linear[5]	[0,0067;4,8517e+008]	---------	5,9656	-----	0,090476±0,03901
Poly	[0,0067;4,8517e+008]	[0,0821;20,1825]	2,8191	0,92632	0,090476±0,03901
RBF	[0,0067;4,8517e+008]	[0,0821;20,1825]	23,9077	1,72008	0,090476±0,03901
ERBF	[0,0067;4,8517e+008]	[0,0821;12,1825]	428,2820	1,5009	0,090476±0,03901
Tangente	[0,0067;4,8517e+008]	[-5,9681;6,1323]	55604,382	0,0732779375	0,090476±0,03901
Spline	[0,0067;4,8517e+008]	---------	1,7353	-----	0,090476±0,03901

Tabela 4.6 – Resultado para busca realizada sobre o conjunto de dados de ALON *et al*. (1999).

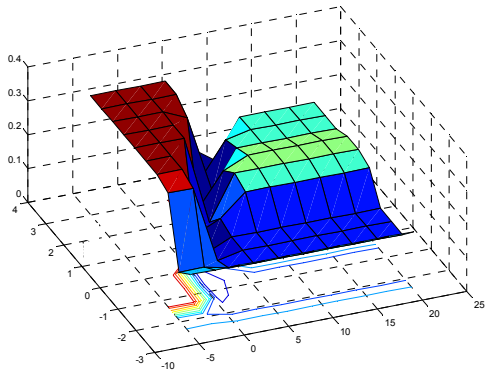


Figura 4. 47 – Superfície de busca para o primeiro nível: problema do câncer de cólon utilizando o kernel RBF.

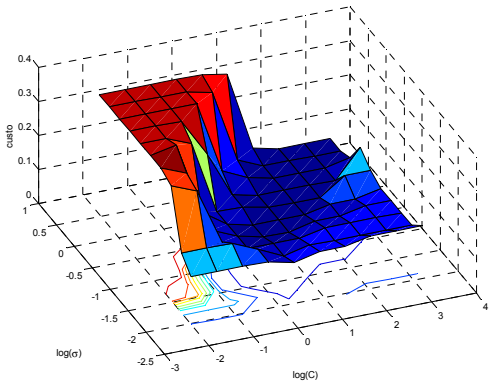


Figura 4.48 – Superfície de decisão para o sesgundo nível: problema do câncer de cólon utilizando o kernel RBF.

Analisando as Figuras 4.47 e 4.48, pode-se concluir que existe uma região onde o SVM não produz resultados muito bons. Por outro lado, podemos perceber uma outra região (um vale) onde o desempenho do SVM foi muito superior. Isto mostra a necessidade de sintonização adequada dos parâmetros para alcançar um bom desempenho em aplicações práticas. Analisando a Tabela 4.6, podemos notar que os diversos *kernels* apresentam o mesmo desempenho, quando o parâmetro *C* e o parâmetro do kernel são sintonizados

[5] Os dados foram normalizados dividindo-se *x* pela norma de *x*.

adequadamente. Os resultados obtidos são equivalentes aos melhores publicados na literatura (GOLUB *et al.*, 1999).

4.21 Seleção de características

Seleção de características e extração de características são duas abordagens relevantes para reduzir a dimensão do vetor de entrada (JAIN *et al.*, 2000). Enquanto a seleção de características refere-se à seleção de características no espaço de medição e as características obtidas são um subconjunto das variáveis de entrada original, na extração de características recorre-se a uma transformação das variáveis de entrada originais e as características fornecidas são um conjunto de novas variáveis no espaço transformado. Geralmente, as características fornecidas pelo método de extração de características podem não ter um significado físico isolado.

Em muitos problemas de aprendizado supervisionado, a seleção de características é importante por uma série de razões: desempenho de generalização, escassez de tempo de execução, restrições e interpretações impostas pelo próprio problema.

Esta seção irá se concentrar em problemas de classificação. Dado um conjunto com N amostras, com entrada $\mathbf{x}_i \in \Re^n$ e saída $y \in \{+1,-1\}$, obtidos de uma distribuição de probabilidade $P(\mathbf{x},y)$, o objetivo é selecionar um subconjunto de características preservando ou mantendo a habilidade de discriminação do classificador. Uma estratégia seria usar "força bruta" para pesquisar sobre todos os possíveis subconjuntos de características. No entanto, este é um problema combinatorial e é preciso levar em conta a qualidade da solução e o custo computacional de qualquer algoritmo empregado.

Em muitas aplicações de classificação de padrões, os dados são representados por vetores de características de alta dimensionalidade. Há duas razões para reduzir a dimensionalidade da representação de um padrão tanto quanto possível. Primeiramente, a representação de baixa dimensão reduz a sobrecarga computacional e pode melhorar a eficácia dos algoritmos de otimização associados à tarefa de implementação do

classificador. Segundo, baixa dimensionalidade tende a melhorar a habilidade de generalização de algoritmos de classificação.

O classificador que maximiza seu desempenho junto aos dados de treinamento pode nem sempre produzir um bom desempenho junto aos dados de teste. Este é o resultado da habilidade de generalização. O desempenho do classificador sobre os dados de teste depende de alguns fatores, incluindo tamanho do conjunto de treinamento, número de características (dimensão da entrada) e a complexidade do classificador.

É difícil estabelecer um relacionamento exato entre o tamanho do conjunto de treinamento e o número de características. Uma regra heurística defendida por JAIN *et al*. (2000) e BERRY & LINOFF (1997) é que, para cada característica, deve-se ter no mínimo 10 exemplos. No entanto, para SVMs o tamanho do conjunto de treinamento pode ser pequeno. SVMs são, portanto, menos susceptíveis a sobre-ajustar os dados que outros algoritmos de classificação e regressão, baseados por exemplo em redes neurais artificiais. No contexto de SVMs, a redução da dimensionalidade é motivada principalmente pela questão da redução da carga de processamento computacional.

O problema de seleção de característica pode ser tratado da seguinte forma:

i. dado um número $k << n$ fixado, encontrar *k* características dentre as *n* disponíveis que produza o menor erro de generalização esperado; ou
ii. dado um erro de generalização esperado máximo ϑ, encontrar o menor *k* possível.

Em ambos os problemas, o erro de generalização esperado é desconhecido e, assim, deve ser estimado. Em WESTON *et al*. (2000) é considerado o problema (*i*). Observe que a escolha de *k* no problema (*i*) pode usualmente ser formulado como o dual do problema (*ii*).

O problema (*i*) é formulado como segue: dado um conjunto fixo de funções $y = f(\mathbf{x}, \boldsymbol{\alpha})$, deseja-se encontrar um pré-processamento dos dados $\mathbf{x} \mapsto (\mathbf{x} * \boldsymbol{\beta}), \boldsymbol{\beta} \in \{0,1\}^n$, e os parâmetros β da função *f* que produzem menor valor de

$$\tau(\beta, \alpha) = \int V(y, f((\mathbf{x} * \boldsymbol{\beta}), \alpha) dP(\mathbf{x}, y) \tag{189}$$

sujeito a $\|\boldsymbol{\beta}\|_0 = k$, onde $P(\mathbf{x},y)$ é desconhecida, $\mathbf{x} * \boldsymbol{\beta} = (x_1\beta_1, \ldots, x_n\beta_n)$ denota um produto ponto a ponto, $V(\cdot,\cdot)$ é um funcional de perda e $\|\cdot\|_0$ é a norma zero.

Baseado no critério utilizado para avaliação do subconjunto de características, métodos de seleção de características podem ser classificados em duas categorias: "métodos de filtro" e "invólucro" (KOHAVI & JOHN, 1997). *Métodos de filtro* são definidos como um passo de pré-processamento para um processo de indução, que pode remover atributos irrelevantes antes que a indução ocorra e, assim, espera-se que seja válido para qualquer conjunto de funções $f(\mathbf{x}, \boldsymbol{\alpha})$. Portanto, um método de filtro emprega propriedades intrínsecas aos dados. Por exemplo, um método de filtro bastante conhecido é o coeficiente de correlação de Pearson. O *método invólucro* (do inglês *wrapper*) é definido como uma busca através do subconjunto do espaço de características usando o desempenho estimado a partir do algoritmo de indução como uma medida de qualidade do subconjunto de características em particular. Assim, aproxima-se $\tau(\boldsymbol{\beta}, \boldsymbol{\alpha})$ por:

$$\tau(\boldsymbol{\beta}, \boldsymbol{\alpha}) = \min_{\boldsymbol{\beta}} \tau_{a\lg}(\boldsymbol{\beta}),$$

sujeito a $\boldsymbol{\beta} \in \{0,1\}^n$ onde $\tau_{a\lg}$ é um algoritmo de aprendizado sobre os dados pré-processados com $\boldsymbol{\beta}$ fixo. O método invólucro pode fornecer soluções com melhor desempenho que os métodos de filtro (KOHAVI, 1995), mas em geral são computacionalmente mais custosos, uma vez que o algoritmo de indução $\tau_{a\lg}$ deve ser avaliado sobre cada conjunto de característica (vetor $\boldsymbol{\beta}$) considerado, tipicamente usando desempenho sobre um conjunto separado como uma medida de qualidade. No caso de classificadores SVM, o método invólucro, apesar de apresentar melhor desempenho que o método de filtro, é computacionalmente custoso, porque o treinamento de um classificador SVM demanda muito esforço computacional se o número de exemplos de treinamento for grande.

Na literatura, poucos algoritmos foram propostos para seleção de característica (BRADLEY *et al.*, 1998; BRADLEY & MANGASARIAN, 1998; WESTON *et al.*, 2001; GUYON *et al.*, 2002). Em BRADLEY *et al.* (1998), foi proposto um método de programação matemática que minimiza uma função côncava sobre um conjunto poliedral. Em BRADLEY & MANGASARIAN (1998), foi realizada uma seleção de um subconjunto de características introduzindo um termo extra na função-objetivo para penalizar o tamanho do subconjunto

de características. WESTON *et al*. (2001) introduziu uma representação vetorial binária para tratar a presença ou não das características para o critério de otimização, com a motivação de aproximação do vetor binário por um vetor de valor real, podendo-se usar o método do gradiente descendente para buscar o valor ótimo do vetor binário e o correspondente subconjunto de características.

Basicamente, os três métodos mencionados acima avaliam características sobre bases individuais, embora as características funcionem verdadeiramente em forma coletiva na função discriminante. Para resolver este problema, foi proposto em GOLUB *et al.* (1999) um algoritmo recursivo de eliminação de características para SVM (SVM RFE), que avalia as características sobre uma base coletiva. O SVM RFE treina uma SVM somente uma vez a fim eliminar uma característica e, portanto, exige menos esforço computacional que um método invólucro.

4.21.1 Exemplos de seleção de característica

Nesta seção, serão apresentados resultados envolvendo o algoritmo SVM RPE aplicado ao problema proposto por ALON *et al.* (1999) e descrito anteriormente. Os parâmetros utilizados serão os mesmos encontrados na seção 4.20 para o problema de seleção de parâmetros. Os resultados para todos os *kernels* considerando a função de perda ε-insensível são apresentados na Tabela 4.7. Analisando a Tabela 4.7, pode-se observar que o desempenho para todos os *kernels* com 100 características é idêntico ao caso com 2000 caracteristicas. Isto significa que, ao invés de utilizar 2000 genes para a tarefa de classificação, poder-se-ia utilizar apenas 100 características, sem queda de desempenho.

Tipo Kernel	Parâmetros		Número de Características				
	C	Param	2000	100	10	5	2
Linear	5,9656	------	0,090476±0,03901	0,090476±0,03901	0,15714±0,039236	0,25000±0,057915	0,64762±0,044868
Poly	2,8191	0,92632	0,090476±0,03901	0,090476±0,03901	0,3381±0,061864	0,35238±0,044868	0,35238±0,044868
RBF	23,9077	1,72008	0,090476±0,03901	0,090476±0,03901	0,15714±0,052669	0,17143±0,046386	0,30476±0,061147
ERBF	428,2820	1,5009	0,090476±0,03901	0,090476±0,03901	0,15714±0,052669	0,17143±0,046386	0,30476±0,061147
Tangente	55604,382	0,0732779375	0,090476±0,03901	0,090476±0,03901	0,19048±0,062894	0,15476±0,065566	0,30000±0,055465
Spline	1,7353	-----	0,090476±0,03901	0,090476±0,03901	0,15714±0,039236	0,22125±0,045613	0,35238±0,044868

Tabela 4.7– Resultado para o problema proposto por ALON *et al.* (1999) com validação cruzada e SVM RPE.

4.22 Considerações finais

SVM é uma abordagem atrativa para modelagem de dados. Esta técnica combina controle da capacidade de generalização com o tratamento da maldição da dimensionalidade. Utiliza as chamadas funções *kernel* para tornar o problema tratável computacionalmente. A formulação tradicional resulta em um problema de otimização quadrática com restrições, que pode ser solucionada por métodos computacionais específicos.

Em problema de classificação, obtém-se controle da capacidade de generalização maximizando a margem, que corresponde à minimização do vetor de pesos na estrutura canônica. A solução é obtida como um conjunto de vetores-suporte que podem ser esparsos. A minimização do vetor de pesos juntamente com uma função de perda pode ser adotada para problemas de regressão.

No decorrer deste capítulo, além das vantagens e do grande potenical de aplicação junto a problemas avançados de classificação e regressão, várias limitações da abordagem SVM foram levantadas, como a escolha da função e dos parâmetros do *kernel*, a escolha do parâmetro C e o problema de escalabilidade quando o número de amostras de treinamento aumenta. Na tentativa de atenuar várias destas limitações, nos Capítulos 5 e 6 serão propostas extensões desta técnica para ensemble e mistura de especialistas, respectivamente, incluindo estudo de casos.

Capítulo 5

Ensembles de Máquinas de Vetores-Suporte

Resumo: Máquinas de Vetores-Suporte (SVMs) possuem alta capacidade de generalização, conforme apresentado no Capítulo 4. No entanto, aspectos estruturais e paramétricos de projeto podem conduzir a uma degradação de desempenho. Tais aspectos dizem respeito principalmente à escolha a priori da função de *kernel* e do parâmetro *C*, sendo a função de *kernel* responsável por realizar o mapeamento não-linear e o parâmetro *C* responsável por definir a importância relativa entre a maximização da margem de separação e a minimização do erro de treinamento. Além disso, a implementação computacional da abordagem SVM recorre a algoritmos aproximados de otimização, os quais, embora viabilizem a aplicação da metodologia, podem conduzir a um desempenho aquém daquele indicado na teoria. Assim, na ausência de uma metodologia sistemática e de baixo custo computacional, ensembles de máquinas de vetores-suporte se apresentam como alternativas capazes de amenizar significativamente os efeitos negativos de especificações equivocadas de projeto. Mais ainda, a existência de configurações estruturais e paramétricas alternativas para os modelos de SVM acaba por sustentar a implementação de comitês de máquinas. Sendo assim, o objetivo deste capítulo será explorar conjuntamente algumas potencialidades advindas da abordagem SVM com aquelas da abordagem ensemble. Várias extensões e novas configurações de ensemble, tendo máquinas de vetores-suporte como componentes, são propostas, implementadas e aplicadas à solução de problemas de

classificação e regressão, possibilitando assim a realização de análises comparativas de desempenho. As contribuições deste capítulo envolvem as três fases associadas à abordagem ensemble: geração, seleção e combinação de modelos. Os principais tópicos a serem abordados são: (*i*) proposição de ensembles de SVMs heterogêneas; (*ii*) proposição de novos critérios de ordenação para seleção de componentes baseados em dimensão VC e margem de separação; (*iii*) proposição de critérios de seleção de componentes baseados em algoritmos genéticos; (*iv*) extensão do conceito de ensemble de múltiplos estágios para SVM; e (*v*) extensão do conceito de ensemble para problemas de classificação com múltiplas classes.

5.1 Introdução

A proposição de ensembles de máquinas de vetores-suporte se apresenta, inicialmente, como uma contradição. Nos Capítulos 2 e 4 foi visto que tanto a abordagem ensemble como a abordagem SVM visam maximizar a capacidade de generalização dos modelos de classificação e regressão. Logo, elas podem ser tomadas como propostas competitivas e, em princípio, não deveria haver ganho de desempenho expressivo com um ensemble, cujos componentes são máquinas de vetores-suporte, quando comparado a uma única máquina de vetores-suporte. Em outras palavras, ensembles produzem ganhos expressivos de desempenho quando seus componentes são modelos instáveis e máquinas de vetores-suporte não se enquadram, em teoria, nesta categoria de modelos.

Entretanto, serão apresentadas a seguir quatro razões pelas quais pode haver ganho de desempenho com ensembles de máquinas de vetores-suporte, sendo que todas elas já foram devidamente tratadas no Capítulo 4:

I. Não existe um procedimento sistemático para a escolha da melhor função *kernel* junto a cada aplicação;
II. Não existe um procedimento sistemático para a escolha do parâmetro *C* (veja seção 4.20) junto a cada aplicação;

III. A implementação computacional de máquinas de vetores-suporte recorre a algoritmos aproximados (BURGES, 1998), de modo que as garantias teóricas de maximização da capacidade de generalização deixam se existir;
IV. Especificamente no caso de problemas de classificação envolvendo múltiplas classes, a sua solução empregando a abordagem SVM vai requerer o emprego de múltiplas máquinas de vetores-suporte (BOTTOU *et al.*, 1994; KNERR *et al.*, 1990) voltadas para classificação binária, podendo levar a uma degradação de desempenho.

Por outro lado, ensembles de redes neurais artificiais (HANHEM & SALAMON, 1990; HASHEM, 1997; SHARKEY, 1999) envolvem a geração, seleção e combinação linear / não-linear de um conjunto de redes neurais individuais implementadas para resolver o mesmo problema. A diversidade em generalização das redes neurais que irão compor o ensemble é patrocinada pela imposição de variação de alguns parâmetros estruturais e/ou de treinamento, conforme mencionado no Capítulo 2. Tal ensemble deverá apropriadamente integrar o conhecimento embutido nas redes neurais componentes, produzindo modelos mais acurados e robustos (BAXT, 1992; TRESP, 2001).

Logo, este cenário para ensembles, originalmente vinculado a redes neurais artificiais, as quais são reconhecidas como modelos instáveis (BREIMAN, 1996b), pode ser estendido para a adoção de máquinas de vetores-suporte como componentes do ensemble. Espera-se que o ensemble de SVMs possa melhorar o desempenho de generalizado em relação a uma única máquina de vetores-suporte, tanto para problemas de classificação como regressão.

As contribuições deste capítulo estão organizadas em quatro itens:

I. Formulação de ensembles de SVMs com *kernels* heterogêneos, isto é, ensembles de SVMs heterogêneas (HE-SVMs, do inglês *heterogeneous ensembles of SVMs*) (LIMA *et al.*, 2002): esta abordagem promove a configuração automática e sintonia da máquina de aprendizado baseada em SVMs com *kernels* distintos, com seleção automática das SVMs que adotam os *kernels* mais apropriados para cada aplicação. As aplicações incluem dados sujeitos a ruído.
II. Proposição de novos critérios de ordenação de classificadores ou estimadores baseados na dimensão VC e na margem de separação. Conforme mencionado no

Capítulo 4, a aplicação do método de seleção baseado em PERRONE & COOPER (1993) requer que os classificadores estejam ordenados segundo algum critério. O critério mais utilizado é o erro quadrático médio. No entanto, tal critério é muito dependente da qualidade do conjunto de seleção. A proposição destes novos critérios tem como objetivo aumentar a robustez do processo de seleção.

III. Utilização de algoritmo genético como uma técnica de busca para encontrar o melhor conjunto de componentes, dentre um conjunto representativo de candidatos. Com esta técnica, substitui-se o critério de seleção de PERRONE & COOPER (1993).

IV. Por fim, a extensão da abordagem de ensembles de múltiplos estágios para SVMs, proposto inicialmente por YANG *et al.*(2002) para redes neurais artificiais.

Este capítulo é organizado como segue: primeiramente, é apresentada a formalização de ensembles de máquinas de vetores-suporte, incluindo uma discussão sobre trabalhos relacionados. Em seguida, as contribuições mencionadas acima serão descritas, seguidas do relato e análise de alguns experimentos realizados. Finalmente, simulações envolvendo todas as abordagens propostas serão apresentadas.

5.2 Ensembles de SVMs

Um ensemble de SVMs é uma união de vários classificadores, ou então regressores, cujas decisões individuais são combinadas de alguma forma para conduzir a uma resposta de consenso, conforme a estrutura apresentada na Figura 5.1. A saída de um ensemble de SVMs, contendo M componentes, é dada por:

$$f(\mathbf{x}) = \sum_{k=1}^{M} f_k(\mathbf{x}) = \sum_{k=1}^{M} \pi_k \left(\mathbf{w}_k^T \varphi_k(\mathbf{x}) + b_k \right), \tag{5.1}$$

onde $\mathbf{w}_k$ e b_k são, respectivamente, os pesos e bias do k-ésimo componente SVM, e π_k é o fator de ponderação de cada componente SVM. Esta formulação é similar àquela proposta por KWOK (1998) no contexto de mistura de especialistas. No entanto, há três diferenças que merecem ser destacadas: (*i*) aqui, toda tarefa é efetuada por meio de redundância e via decomposição funcional; (*ii*) os π_k podem ser ou não uma função de $\mathbf{x}$; e (*iii*) nesta

formulação, cada componente SVM possui uma função *kernel* distinta, ou seja, $\varphi_i \neq \varphi_j$ para $i \neq j$.

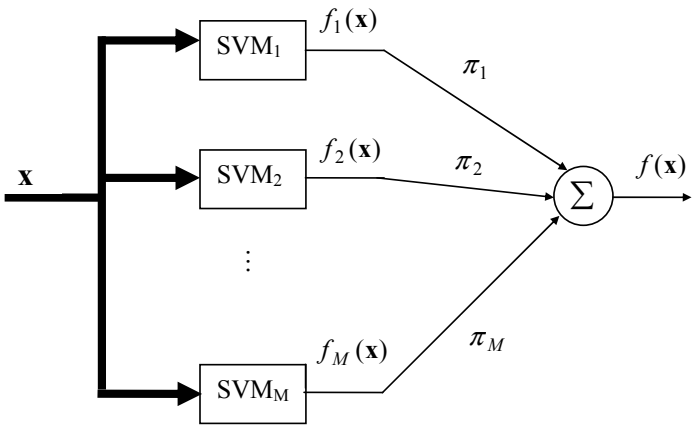


Figura 5.1 – Arquitetura de um ensemble de SVMs

Enquanto uma arquitetura típica de SVM opera com somente uma configuração de máquina de aprendizado no espaço de características de alta dimensionalidade, um ensemble de SVMs utiliza uma combinação de *M* modelos de SVM (LIMA *et al*., 2002), os quais diferem entre si, por exemplo, na escolha na função *kernel* e/ou do parâmetro *C*. Um dos objetivos básicos desta iniciativa é fornecer um mecanismo automático que retira do projetista a difícil tarefa de escolher, por exemplo, a melhor função *kernel* para cada aplicação. Além disso, justamente devido ao emprego de diferentes *kernels*, cria-se condições de combinar múltiplas soluções alternativas, com possíveis ganhos de desempenho frente à melhor solução individual.

O preço a pagar é evidente: é necessário resolver o mesmo problema tantas vezes quantos forem os candidatos a comporem o ensemble, além da necessidade de alocar recursos computacionais para realizar a seleção e combinação de modelos. Outro aspecto relevante diz respeito ao número *M* de componentes. Como visto no Capítulo 2, quanto maior *M*, menor é a possibilidade de divergência em generalização. Sendo assim, os processos de seleção normalmente alocam menos que 10 componentes para o ensemble, mesmo que existam dezenas disponíveis. Na presença de um método de seleção de

modelos, o número de candidatos gerados é sempre maior ou igual ao número de componentes do ensemble.

5.3 Trabalhos Relacionados

Além da abordagem apresentada nesta tese, outras abordagens têm também buscado a inclusão da metodologia SVM em comitês de máquinas. Neste contexto, KIM *et al.* (2002) apresentou a idéia de ensemble de SVMs, contendo resultados de simulações envolvendo problemas de classificação binária e com múltiplas classes. Eles empregaram técnicas diferentes para agregar os componentes baseados em SVMs. O treinamento de cada SVM é realizado independentemente por meio de um conjunto de dados produzido via *bagging*. Entretanto, eles não trataram o problema de regressão e nem a questão relacionada à escolha da função *kernel*.

Explorando também o conceito de re-amostragem adaptativa dos dados de treinamento, PAVLOV *et al.* (2000) aplicaram o algoritmo *boosting* para a geração de ensembles de SVMs. A principal motivação dos pesquisadores era aumentar a velocidade de convergência do algoritmo SVM proposto por PLATT (1998), isto é, o algoritmo de otimização por minimização seqüencial (SMO, do inglês *sequencial minimal optimization*) (CRISTIANINI & SHAWE-TAYLOR, 2000; SCHÖLKOPF & SMOLA, 2002). Entretanto, os resultado apresentados via Boost-SMO, como chamaram sua abordagem, foram relativamente os mesmos alcançados pelo SMO convencional.

Por outro lado, o comitê bayesiano de SVMs (BC-SVM) (SCHWAIGHOFER & TRESP, 2001) foi concebido tendo em mente a questão da escalabilidade do treinamento de cada SVM. Nesta metodologia, o conjunto de dados de treinamento é particionado em vários conjuntos de dados menores e as estimativas dos componentes individuais são combinadas usando uma forma de ponderação baseada na covariância e em uma perspectiva bayesiana (TRESP, 2001). O algoritmo resultante para treinamento das SVMs escala linearmente com o número de exemplos de treinamento. Entretanto, tanto Boost-SMO como BC-SVM não apresentaram um nível de desempenho superior quando comparados com a implementação

convencional SVM. Uma possível razão para isso seria a utilização da mesma função *kernel* para todos os componentes.

VALENTINI & DIETTERICH (2002) investigaram o método de decomposição bias-variância para um ensemble de SVMs, cujo objetivo era descobrir o relacionamento dos *kernels* e seus parâmetros no processo de aprendizado de cada SVM. No entanto, tais análises bias-variância foram realizadas somente sobre variações nos parâmetros do *kernel*, a fim de produzir um ensemble de SVMs dotadas de um mesmo tipo de *kernel*. Também foi empregado *bagging*. Os resultados apresentados já indicaram ganhos significativos de desempenho.

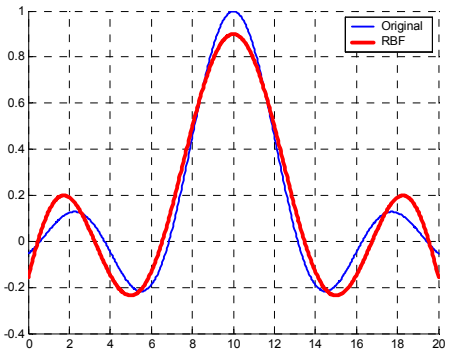
Nesta tese, a associação entre a abordagem SVM e comitês de máquinas é vista sob outra perspectiva, com ênfase na combinação de máquinas de vetores-suporte dotadas de diferentes funções de *kernel*, visando superar ou, ao menos, amenizar algumas das limitações da abordagem SVM e produzir ferramentas com melhor desempenho de generalização.

As próximas 4 seções descrevem as principais contribuições relacionadas ao tema do capítulo.

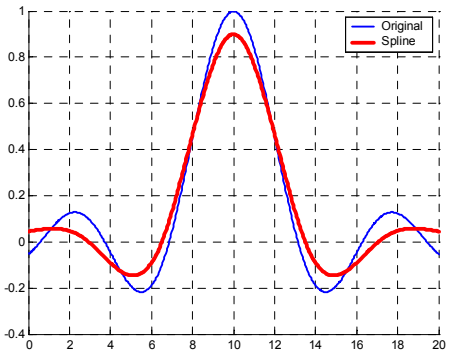
5.4 Ensembles de SVMs Heterogêneas (HE-SVMs)

Reconhecendo o papel importante das funções *kernel* como medida de similaridade para a construção do espaço de características de alta dimensionalidade, ensembles de SVMs heterogêneas (HE-SVMs) diferem das abordagens já propostas na literatura, visto que aqui cada componente $f_k(\mathbf{x})$ é uma instância de uma máquina de vetores-suporte com um tipo de *kernel* diferente. Desta forma, medidas de similaridade distintas (associadas à concepção do espaço de características) podem trabalhar cooperativamente no processo de extração de características, a fim de melhorar o mapeamento de classificação ou regressão do HE-SVMs. Como apontado por SMITS & JORDAAN (2002), a combinação de *kernels* parece ser uma estratégia promissora, uma vez que esta tenta agrupar as qualidades exibidas por cada *kernel* individualmente. Com isso, pode resultar uma solução de consenso que privilegie simultaneamente as habilidades de interpolação e extrapolação. Entretanto, ao

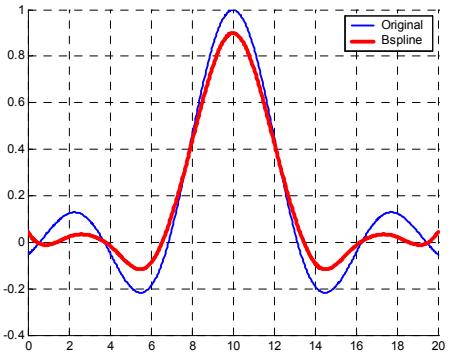
invés de fazer uso de um ensemble, SMITS & JORDAAN (2002) produziram uma nova função de *kernel* por meio de uma mistura aditiva (combinação convexa) de *kernels*, explorando as peculiaridades de cada um em regiões diferentes do espaço de entrada.



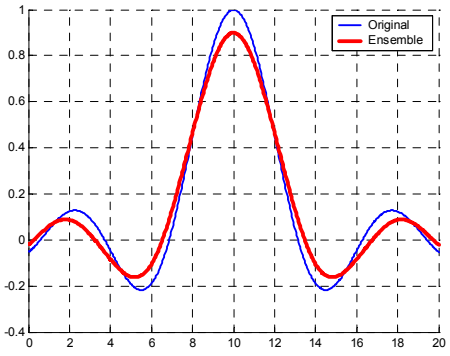
(*a*): *Kernel* RBF



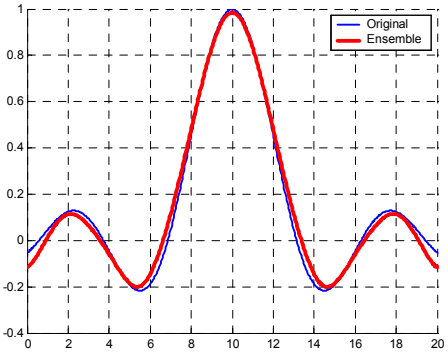
(*b*): *Kernel* Spline



(*c*): *Kernel* B-spline



(*d*): Ensemble com média simples



(*e*):Ensemble com média ponderada

Figura 5.2 – Análise comparativa evidenciando o efeito do uso de *kernels* distintos e o ganho de desempenho proporcionado pelos ensembles.

A fim de confirmar nosso argumento, as Figuras 5.2 (*a*)-(*c*) ilustram o comportamento de três SVMs com funções *kernel* diferentes aplicadas a um problema de regressão, envolvendo a função sinc unidimensional (veja $g^{(1)}$ no Apêndice A). É possível verificar que os *kernels* apresentam padrões de erro distintos em diferentes regiões do espaço de regressão. Quando estes *kernels* são agrupados em um ensemble, o desempenho do modelo resultante tende a ser melhor, uma vez que a combinação produz uma solução de consenso que possivelmente elimina as distorções indevidas (veja Figuras 5.2 (*d*)-(*e*)). Também se mostra significativo o fato de que critérios de combinação diferentes podem produzir desempenhos diferentes para um mesmo conjunto de componentes (veja Figuras 5.2 (*d*)-(*e*)).

Em termos de implementação, o HE-SVMs inclui os seguintes passos: (*i*) treinamento dos componentes, ou seja, síntese das SVMs; (*ii*) seleção de componentes mais indicados a compor o ensemble (de acordo com os métodos apresentados no Capítulo 2); e (*iii*) cálculo dos pesos da combinação para os componentes selecionados.

Na seqüência, será apresentada uma formulação do problema de treinamento do HE-SVMs com base em uma formulação QP (do inglês *quadratic programming*).

5.4.1 Problema QP para HE-SVMs

A fim de formalizar o processo de treinamento do HE-SVMs como um problema de programação quadrática (QP), será adotada uma estratégia similar àquela realizada no Capítulo 4. Considere o seguinte problema:

$$\min \frac{1}{2}\sum_{k=1}^{M}\|\mathbf{w}_k\|^2 + C\sum_{i=1}^{N}\left(\xi_i + \xi_i^*\right), \tag{5.2}$$

sujeito a

$$\begin{cases} y_i - \sum_{k=1}^{M}\pi_{ki}\left(\mathbf{w}_k^T\varphi_k(\mathbf{x}_i) + b_k\right) \le \varepsilon + \xi_i^*, \\ \sum_{k=1}^{M}\pi_{ki}\left(\mathbf{w}_k^T\varphi_k(\mathbf{x}_i) + b_k\right) - y_i \le \varepsilon + \xi_i, \\ \xi_i, \xi_i^* \ge 0 \quad \text{para} \quad i = 1, \cdots, N. \end{cases} \tag{5.3}$$

Então, o problema de regressão QP resultante pode ser escrito como:

$$\max_{\alpha,\alpha^*} L(\alpha,\alpha^*) = \max_{\alpha,\alpha^*} \varepsilon\sum_{i=1}^{N}\left(\alpha_i^* + \alpha_i\right) + \sum_{i=1}^{N} y_i\left(\alpha_i^* - \alpha_i\right) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{M}(\alpha_i^* - \alpha_i)(\alpha_j - \alpha_j)\pi_{ki}\pi_{kj}K_k(\mathbf{x}_i,\mathbf{x}_j), \quad (5.4)$$

sujeito a (*i*) $\sum_{i=1}^{N}\left(\alpha_i^* - \alpha_i\right)\pi_{ki} = 0$, para $k = 1,\cdots,M$; e (*ii*) $0 \le \alpha_i, \alpha_i^* \le C$, para $i = 1,\cdots,N$.

Comparando a formulação proposta no Capítulo 4 para regressão com a formulação (5.4), pode-se afirmar que, exceto pelo termo adicional $\sum_{k=1}^{M}\pi_{ki}\pi_{kj}$ e uma simples modificação no conjunto de restrição linear, os problemas QP são equivalentes.

Analisando agora o caso do problema de classificação, considerando o espaço de entrada linearmente separável, pode-se definir o seguinte problema:

$$\min \frac{1}{2}\sum_{k=1}^{M}\|\mathbf{w}_k\| + C\sum_{i=1}^{N}\xi_i, \quad (5.5)$$

sujeito a

$$\begin{cases} y_i\left(\sum_{k=1}^{M}\pi_{ki}\left(\mathbf{w}_k^T\varphi_k(\mathbf{x}_i) + b_k\right)\right) \ge 1 - \xi_i \\ \xi_i \ge 0 \text{ para } i = 1,\cdots,N. \end{cases} \quad (5.6)$$

O problema QP de classificação resultante pode ser especificado na forma:

$$\max L(\alpha,\alpha^*) = \sum_{i}^{N}\alpha_i - \frac{1}{2}\sum_{i,j,k}^{N} y_i y_j \alpha_i \alpha_j \pi_{ki}\pi_{kj}K_k(\mathbf{x}_i,\mathbf{x}_j) \quad (5.7)$$

sujeito a (*i*) $\sum_{i=1}^{N}\alpha_i y_i \pi_{ki} = 0$, para $k = 1,\cdots,M$; (*ii*) $0 \le \alpha_i \le C$, para $i = 1,\cdots,N$. Há aqui também há uma equivalência em relação ao caso de uma única SVM e a formulação (5.7).

Apesar dos equacionamentos (5.4) e (5.7) serem bastante simples e empregarem somente um problema QP envolvendo todos os tipos de *kernel* $K_k(\mathbf{x}_i,\mathbf{x}_j)$ para $k = 1,\cdots,M$, eles apresentam uma desvantagem em relação à dimensão da matriz de *kernels* **K**. A matriz **K** tem uma dimensão que depende do número de dados de treinamento e o número de componentes a serem utilizados no ensemble. Com o intuito de superar tal desvantagem, várias simulações foram realizadas comparando o desempenho do ensemble formado com SVMs treinadas simultaneamente, utilizando os resultados acima, e treinadas independentemente, utilizando a formulação do Capítulo 4. Os resultados se mostraram equivalentes. Assim, nos experimentos realizados mais adiante, ainda neste capítulo, será

adotado o treinamento independente para todas as SVMs, embora esta metodologia represente uma solução aproximada.

5.4.2 Resultados Experimentais

Nesta seção, o HE-SVMs é avaliado em relação a cinco problemas de aproximação de funções sem ruído e dois experimentos de classificação de padrões. Para cada análise, três conjuntos de amostras foram definidos, todos derivados da mesma distribuição estatística: um para treinamento, outro para seleção dos componentes e cálculo dos pesos da combinação, e outro para testar o desempenho do HE-SVMs.

O algoritmo utilizado para a abordagem HE-SVMs é apresentado na Figura 5.3.

1. Geração dos componentes;
   - 1.1. Normalize os dados de entrada de acordo com cada *kernel*;
   - 1.2. Treine cada componente SVM associado a *kernels* diferentes empregando o conjunto de treinamento;
2. Seleção de componentes;
   - 2.1. Calcule a saída de cada componente, usando o conjunto de dados de seleção;
   - 2.2. Ordene os componentes de acordo com algum critério;
   - 2.3. Selecione os componentes via o critério baseado em PERRONE & COOPER (1993), apresentado no Capítulo 2;
3. Combinação de componentes;
   - 3.1. Para problemas de classificação,
     - 3.1.1. Utilize um dos critérios de combinação apresentados no Capítulo 2. Normalmente, é empregado o voto majoritário.
   - 3.2. Para problemas de regressão;
     - 3.2.1. Calcule os pesos da combinação (π's) usando o método EQM-OLC (veja Capítulo 2);
4. Avalie o HE-SVMs junto ao conjunto de teste

Figura 5.3 – Algoritmo do HE-SVMs.

Dentre os *kernels* mencionados no Capítulo 4, nesta seção serão utilizados apenas alguns deles, a saber: (*i*) linear, (*ii*) polinomial, (*iii*) função gaussiana de base radial (RBF),

(*iv*) função exponencial de base radial (ERBF), (*v*) tangente hiperbólica, (*vi*) série de Fourier, (*vii*) Splines e (*viii*) B-splines. Uma discussão detalhada sobre estes *kernels* pode ser encontrada em GUNN (1998). Para todos os experimentos realizados, exceto para o experimento de classificação tratando com um problema de bioinformática (no qual um método para ajuste dinâmico dos parâmetros do *kernel* é utilizado), os parâmetros do *kernel* utilizados são aqueles definidos usualmente na literatura e que foram mencionados no Capítulo 4.

Os experimentos a seguir foram divididos em regressão e classificação de padrões. Para regressão, foram definidas funções $g^{(k)}(x_1,x_2)$ (k = 1,...,5) a serem aproximadas, cujas expressões e esboço gráfico são apresentados no Apêndice A. A motivação para considerar apenas duas entradas é a possibilidade de uma visualização 3D dos resultados.

5.4.2.1 Experimentos envolvendo regressão

Na literatura, a função $g^{(1)}$: $[0,20] \rightarrow \Re$ foi empregada por VAPNIK (1995) e as funções $g^{(2)}$, $g^{(3)}$, $g^{(4)}$ e $g^{(5)}$ ($g^{(k)}$: $[0,1]^2 \rightarrow \Re$, para k = 2,...,5) foram utilizadas por HWANG *et al.* (1994), em outros contextos. Estas cinco funções passaram a ser consideradas por outros autores como um "campo de provas" desafiador para problemas de regressão.

As funções $g^{(k)}(\mathbf{x})=g^{(k)}(x_1,x_2)$ (k = 2,...,5) foram supostas desconhecidas de forma que somente amostras obtidas de modo a cobrir apropriadamente o espaço de aproximação estavam disponíveis. O tamanho do conjunto de dados de treinamento (N_{tr}) e seleção (N_{se}) varia de acordo com o experimento realizado. No entanto, o tamanho do conjunto de dados de teste N_{te} foi fixado em 625 amostras para todos os problemas, uniformemente distribuídas. A qualidade da solução obtida foi avaliada comparando-se (via EQM) os valores da saída produzida pelos modelos de regressão considerados com a saída desejada presente no conjunto de dados de teste.

O valor do parâmetro C foi adotado igual a infinito em todos os experimentos, o que implica que, no espaço de características, nenhuma amostra vai estar a uma distância maior que ε do hiperplano ótimo (veja Capítulo 4, seção 4.15).

5.4.2.2 Experimento no. 1

Neste experimento, foram adotados ε = 0,1 e $C = \infty$, (veja Capitulo 4, seção 4.20) e um conjunto de dados de treinamento com 121 amostras. A Tabela 5.1 mostra os resultados encontrados, consistindo de: (*i*) EQM e tipo de *kernel* da SVM com melhor desempenho, em cada caso; (*ii*) EQM e tipos de *kernel* selecionados para compor o HE-SVMs; e (*iii*) porcentagem de ganho (PoG) que é obtida comparando-se o desempenho relativo do HE-SVMs com a melhor SVM. PoG é dado por 1 menos a razão entre o EQM encontrado pelo melhor método de combinação HE-SVMs, média simples ou média ponderada, e o EQM da melhor SVM tomado isoladamente. Verificando esta comparação, é possível averiguar que o HE-SVMs invariavelmente produz uma melhora significativa sobre a melhor SVM tomada isoladamente.

	N_{se}	MELHOR SVM		MELHOR HE-SVM			PORCENTAGEM DE GANHO (%)
		TIPO DE *KERNEL*[1]	EQM	*KERNELS* USADOS	MÉDIA SIMPLES	MÉDIA PONDERADA	
$g^{(1)}$	225	[vii]	0,00566	[vii; iii; vi]	0,00481	0,00421	25,61
$g^{(1)}$	625	[vii]	0,00566	[vii; iii; vi]	0,00481	0,00420	25,79
$g^{(2)}$	225	[vi]	0,00661	[vi; viii; vii; iii]	0,00513	0,00149	77,45
$g^{(2)}$	625	[vi]	0,00661	[viii; vi; vii; iii]	0,00513	0,00147	77,76
$g^{(3)}$	225	[vi]	0,00578	[vi]	0,00578	0,00486	15,91
$g^{(3)}$	625	[vi]	0,00578	[vi]	0,00578	0,00486	15,91
$g^{(4)}$	225	[viii]	0,00708	[viii; vii; vi]	0,00513	0,00349	50,70
$g^{(4)}$	625	[viii]	0,00708	[viii; vii; vi]	0,00513	0,00342	51,69
$g^{(5)}$	225	[viii]	0,00910	[viii; vii; iii]	0,00612	0,00585	35,71
$g^{(5)}$	625	[viii]	0,00910	[viii; iii]	0,00612	0,00590	35,16

Tabela 5.1 – Resultados de teste do Experimento no. 1: regressão (N_{tr}=121)

[1] (*i*) linear, (*ii*) polinomial, (*iii*) função gaussiana de base radial (RBF), (*iv*) função exponencial de base radial (ERBF), (*v*) tangente hiperbólica, (*vi*) série de Fourier, (*vii*) Splines, (*viii*) B-splines (veja seção 5.4)

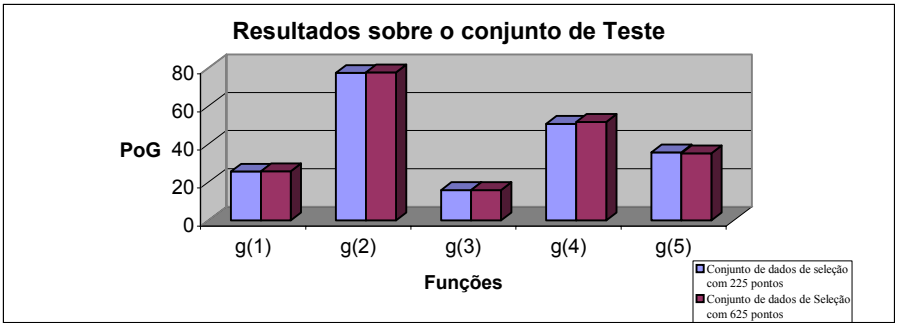


Figura 5.4 – Porcentagem de ganho junto aos 5 problemas de regressão, com dois conjuntos distintos de dados de seleção ($\varepsilon = 0{,}1$, $C = \infty$ e $N_{tr} = 121$)

5.4.2.3 Experimento no. 2

Neste experimento, foi mantido o mesmo conjunto de dados de treinamento usado no experimento anterior, mas o valor de ε foi alterado para 0,01 (dez vezes menor). O objetivo desta alteração é mostrar que, para um conjunto de treinamento pequeno e um valor pequeno de ε, a utilização do melhor SVM pode incorrer em sobre-ajuste dos dados de treinamento, conseqüentemente diminuindo a habilidade de generalização. Em tal caso, o HE-SVMs tende ainda a produzir uma boa capacidade de generalização. Analisando a Tabela 5.7, pode-se notar que o HE-SVMs apresenta uma capacidade de generalização relativamente melhor (exceto para $g^{(4)}$) que aquele apresentado na Tabela 5.1 (a medida PoG ratifica tal fato). Isto é esperado, pois a dimensão VC de cada componente SVM tende a aumentar quando se adota um valor menor para ε. A diminuição da PoG para $g^{(4)}$ pode ser atribuída à diminuição do número de componentes com erros descorrelacionados, uma vez que no, experimento anterior, o método de seleção selecionou 3 componentes e neste, apenas 2 componentes.

	N_{se}	MELHOR SVM		MELHOR HE-SVMS			PORCENTAGEM DE GANHO (%)
		TIPOS DE *KERNEL*[2]	EQM	*KERNELS* USADOS	MÉDIA SIMPLES	MÉDIA PONDERADA	
$g^{(1)}$	225	[iii]	0,000064	[iii; vii; viii]	0,000047	0,000026	59,37
$g^{(1)}$	625	[iii]	0,000064	[iii; vii; viii]	0,000047	0,000026	59,37
$g^{(2)}$	225	[vi]	0,000066	[vi; iii; viii]	0,000037	0,000013	80,30
$g^{(2)}$	625	[vi]	0,000066	[vi; iii; viii]	0,000037	0,000014	78,78
$g^{(3)}$	225	[vi]	0,000072	[vi; iii]	0,000048	0,000035	51,38
$g^{(3)}$	625	[vi]	0,000072	[vi; iii]	0,000048	0,000035	51,38
$g^{(4)}$	225	[vii]	0,001891	[vii; iii]	0,001619	0,001337	29,29
$g^{(4)}$	625	[vii]	0,001891	[vii; iii]	0,001619	0,001331	29,61
$g^{(5)}$	225	[iii]	0,002230	[iii; viii; iv]	0,001410	0,001156	48,16
$g^{(5)}$	625	[iii]	0,002230	[viii; iii; iv]	0,001410	0,001185	46,86

Tabela 5.2 – Resultados de teste do Experimento no. 2: regressão (N_{tr}=121)

Como um resultado não apresentado na Tabela 5.2, para $g^{(4)}$ e $g^{(5)}$, o ensemble resultante teve um desempenho pior que a melhor SVM, isso para os dados de treinamento. Entretanto, uma vez que o cálculo dos pesos de agregação para o ensemble leva em consideração os dados de validação, tal problema de sobre-ajuste tende a ser aliviado, levando assim o ensemble a superar o melhor componente isolado.

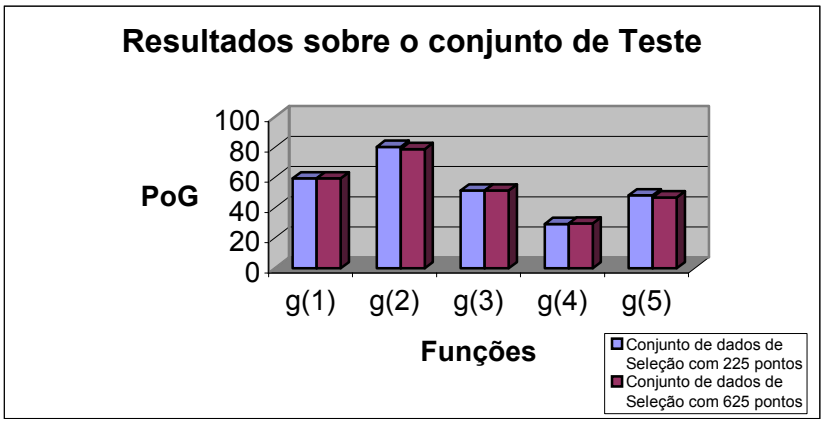


Figura 5.5 – Porcentagem de ganho junto aos 5 problemas de regressão, com dois conjuntos distintos de dados de seleção ($\varepsilon = 0{,}01$, $C = \infty$ e $N_{tr} = 121$)

[2] veja seção 5.4.2

5.4.2.4 Experimento no. 3

Neste experimento, foram mantidos todos os parâmetros do primeiro experimento, exceto o conjunto de dados de treinamento, que foi alterado de 121 pontos para 225 pontos. O propósito foi investigar o comportamento do HE-SVMs considerando a questão da escalabilidade, isto é, o efeito de uma maior densidade amostral junto a um problema de regressão. Repare que, quanto maior a densidade amostral, melhor tende a ser a capacidade de generalização de modelos de regressão isolados, o que não impede que o ensemble melhore a sua capacidade de generalização proporcionalmente, ao combinar modelos isolados. Novamente, como evidenciado na Tabela 5.3, o HE-SVMs produziu desempenho superior para todas as funções em relação a uma única SVM, que é comparável aos resultados do Experimento no. 1. Desta forma, o HE-SVMs mostrou robustez frente à alteração da densidade amostral.

	N_{se}	MELHOR S-SVM		MELHOR HE-SVM			PORCENTAGEM DE GANHO (%)
		TIPO DE *KERNEL*	EQM	*KERNELS* USADOS	MÉDIA SIMPLES	MÉDIA PONDERADA	
$g^{(1)}$	225	[vii]	0,005689	[vii; iii; vi]	0,004873	0,004313	24,18
$g^{(1)}$	625	[vii]	0,005689	[vii; iii; vi]	0,004873	0,004308	24,27
$g^{(2)}$	225	[viii]	0,005682	[vi; viii; vii]	0,004585	0,001092	80,78
$g^{(2)}$	625	[viii]	0,005682	[viii; vi; vii]	0,004585	0,001076	81,06
$g^{(3)}$	225	[vi]	0,005685	[vi; iii]	0,004226	0,004010	29,46
$g^{(3)}$	625	[vi]	0,005685	[vi; iii]	0,004226	0,003991	29,79
$g^{(4)}$	225	[viii]	0,005654	[viii; iii]	0,004399	0,003513	37,86
$g^{(4)}$	625	[viii]	0,005654	[viii; iii]	0,004399	0,003483	38,39
$g^{(5)}$	225	[iii]	0,006621	[viii; vii; iii]	0,004491	0,004385	33,77
$g^{(5)}$	625	[iii]	0,0066216	[viii; iii; vii]	0,004491	0,004297	35,10

Tabela 5.3 – Resultados de teste do Experimento no. 3: regressão (N_{tr}=225)

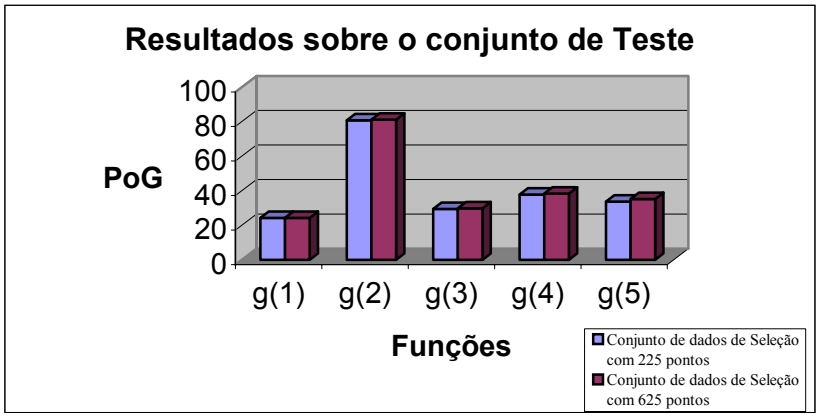


Figura 5.6 – Porcentagem de ganho junto aos 5 problemas de regressão, com dois conjuntos distintos de dados de seleção ($\varepsilon = 0{,}1$, $C = \infty$ e $N_{tr} = 225$)

Um sumário de todos os PoG obtidos encontra-se na Figura 5.7, evidenciando que a abordagem HE-SVMs foi sempre capaz de superar o desempenho do melhor componente tomado isoladamente.

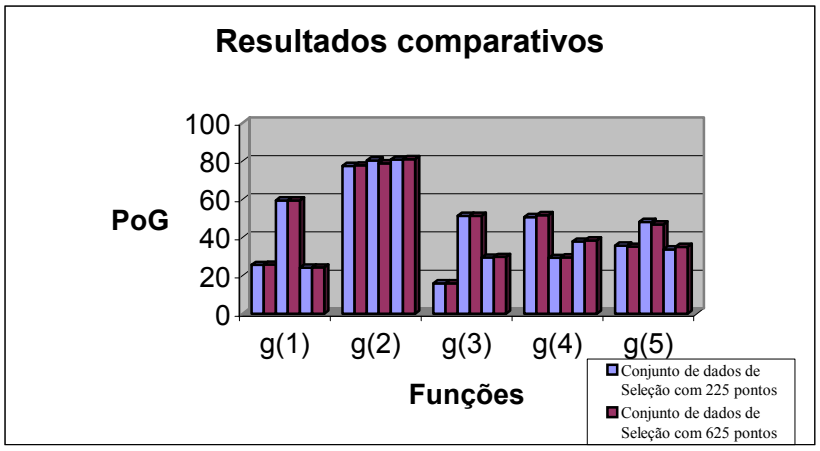


Figura 5.7 – Quadro geral para os 5 problemas de regressão

5.4.3 Classificação de padrões

Nesta seção, será aplicado o HE-SVMs a dois problemas de classificação de padrões. O primeiro é um problema clássico, conhecido como problema das duas espirais entrelaçadas, e o segundo é um problema de bioinformática cujo objetivo é discriminar

entre pacientes portadores de células normais e pacientes portadores de células cancerígenas, considerando milhares de valores de expressão gênica vinculados a experimentos de microarray.

5.4.3.1 Espirais entrelaçadas

O problema das duas espirais foi tratado inicialmente por LANG & WITBROCK (1988), o qual é ilustrado na Figura 5.8. Este consiste de dois anéis entrelaçados, cada um representando uma classe, cujas equações são apresentadas no Apêndice A. A idéia é categorizar os padrões de entrada como pertencendo a uma das espirais (50% dos padrões pertencem a cada classe).

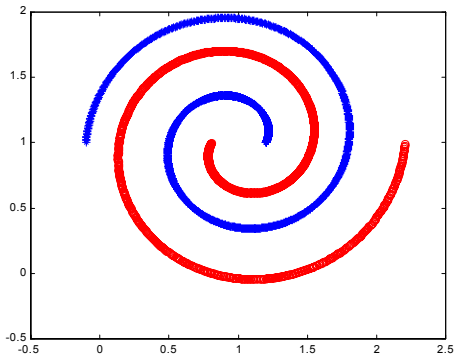


Figura 5.8 – O problema de classificação das duas espirais.

Na literatura, já existem várias abordagens empregando a abordagem SVM para este problema, tal como o trabalho de SUYKENS & VANDEWALLE (1999). Diferente destas abordagens, o objetivo desta seção é averiguar o comportamento de uma SVM isolada e da abordagem HE-SVMs com relação a um conjunto de treinamento muito restrito, ou seja, pobre em amostras, e um conjunto de teste rico em amostras. Análogo à seção anterior, três conjuntos foram gerados: treinamento, seleção e teste, todos amostrados com distribuição uniforme.

Algumas simulações foram realizadas variando o conjunto de dados de treinamento. A Tabela 5.4 apresenta os resultados mais expressivos, apontando o número de padrões classificados incorretamente tanto para o treinamento quanto para o teste, para cada tipo de

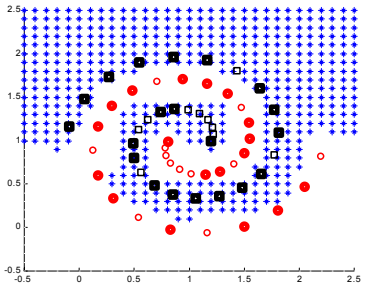
kernel. Além disso, é apresentado o número de vetores-suporte associado (NSV). A coluna com valores em negrito indica o SVM que produziu melhor resultado. Os resultados apresentados foram obtidos para um conjunto de seleção composto de 384 amostras e um conjunto de teste com 944 amostras.

		NÚMERO DE PADRÕES CLASSIFICADORS INCORRETAMENTA PARA CADA SVM								HE-SVM
		TIPO DE *KERNEL*								
		LINEAR (*i*)	POLY (*ii*)	RBF (*iii*)	ERBF (*iv*)	HYP. TANG. (*v*)	FOURIER (*vi*)	SPLINE (*vii*)	BSPLINE (*viii*)	
N_{tr} = 60	Treinamento	24	23	0	0	0	3	0	**0**	0
	Teste	374	367	15	0	89	111	32	**0**	0
	Nº SV	60	60	31	60	51	45	36	**38**	[viii]
N_{tr} = 48	Treinamento	18	18	0	0	2	4	0	**0**	0
	Teste	374	366	131	4	239	272	95	**0**	0
	Nº SV	48	48	23	48	41	36	41	**36**	[viii]
N_{tr} = 44	Treinamento	18	17	0	**0**	0	3	0	0	0
	Teste	379	369	133	**22**	201	386	99	30	0
	NSV	44	44	30	**44**	33	32	34	31	[viii]
N_{tr} = 40	Treinamento	16	17	0	0	0	4	0	**0**	0
	Teste	374	366	172	11	138	360	60	**10**	0
	NSV	40	40	25	40	31	32	29	**30**	[iv,viii,i]
N_{tr} = 36	Treinamento	14	14	0	0	0	10	0	**0**	0
	Teste	374	364	113	0	213	383	109	**0**	0
	NSV	36	36	23	36	30	28	27	**30**	[iv]
N_{tr} = 32	Treinamento	14	13	0	**0**	0	8	0	0	0
	Teste	376	366	223	**23**	287	352	97	76	23
	NSV	32	32	20	**32**	27	29	26	28	[iv]

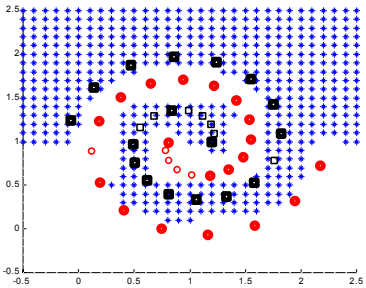
Tabela 5.4 – Resultados de teste para o problema das duas espirais

Analisando os resultados, pode-se observar que $N_{tr} = 36$ pode ser considerado como um limiar, abaixo do qual nenhum modelo é capaz de apresentar erro de teste nulo, nem mesmo o ensemble.

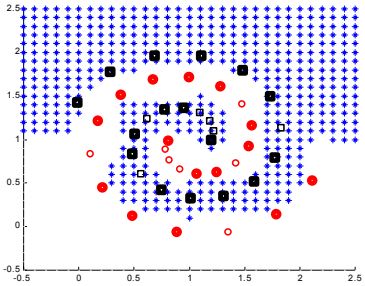
Outro limiar importante é $N_{tr} = 48$, pois para valores acima destes há pelo menos um componente que produz 100% de classificação correta para os dados de teste. Neste caso, o HE-SVMs mostrará apenas desempenho equivalente ao melhor SVM. A Figura 5.9 traz a superfície de decisão apresentada pelo HE-SVMs para todos os valores de N_{tr}, com seus respectivos *kernels* selecionados.



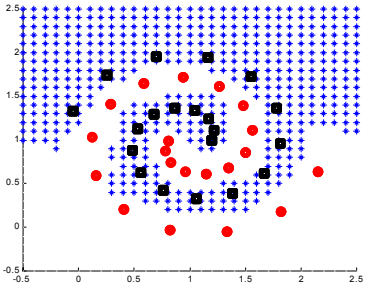
(*a*) N_{tr} = 60, *Kernel* do tipo *viii*



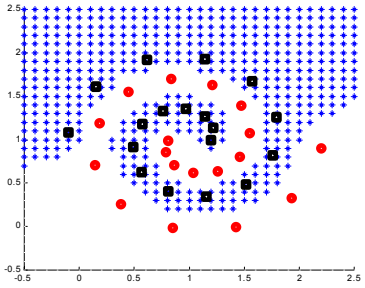
(*b*) N_{tr} = 48, *Kernel* do tipo *viii*



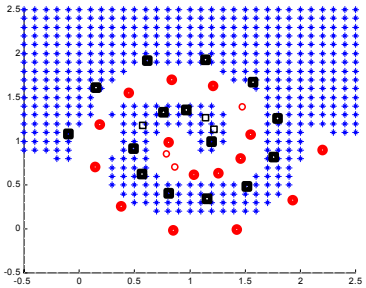
(*c*) N_{tr} = 44, *Kernel* do tipo *viii*



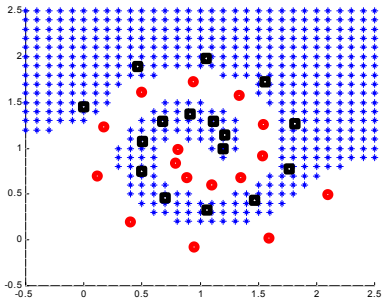
(*d*) N_{tr} = 40, *Kernel* do tipo *iv*, *viii*, *i*



(*e*) N_{tr} = 36, *Kernel* do tipo *iv*



(f) N_{tr} = 36, *Kernel* do tipo *viii*



(*g*) N_{tr} = 32, *Kernel* do tipo *iv*

Figura 5.9 – Resultados para o problema de classificação das duas espirais utilizando HE-SVMs, onde os dados de treinamento são representados por um quadrado preto para uma classe e por um círculo em vermelho para a outra classe. Os pontos para o quais os quadrados ou círculos estão destacados são os vetores-suporte.

Para N_{tr}=36, o HE-SVMs resultante é formado somente por um tipo de *kernel* e todos os exemplos de treinamentos foram considerados como vetores-suporte. Ambos os *kernels viii* e *iv* encontraram 100% de classificação correta (Tabela 5.4). Isto não é verdade para $N_{tr} = 40$, onde todos os componentes produziram classificação incorreta em pelo menos 10% dos padrões. No entanto, com $N_{tr} = 40$ o HE-SVMs resultante produziu 100% de classificação correta, sendo que o ensemble é formado pelos componentes com *kernels iv*, *viii*, e *i* (seguindo a ordem pela qual eles foram selecionados). Todos os exemplos de treinamento foram tomados como vetores-suporte. Mesmo produzindo classificação incorreta, a contribuição do *kernel i* foi importante para dados de classificação localizados na borda extrema (como pode ser visto, muitos dos vetores-suporte produzidos em todos os experimentos residem na região central, pois lá as curvas são mais entrelaçadas). Acredita-se que, com o emprego de outros critérios de combinação, tais como voto majoritário ponderado, este tipo de *kernel* poderá ser excluído do ensemble final, uma vez que a contribuição dos *kernels* restantes deverão ser melhor definida pela ponderação distinta entre os classificadores componentes (LIMA *et al.*, 2002).

Analisando a Tabela 5.4, parece contraditório que, para $N_{tr} = 36$, o HE-SVMs consegue melhor desempenho que para $N_{tr} = 40$. Mas a explicação para este fato está no posicionamento das amostras de treinamento, já que as 36 amostras de um caso não estão contidas nas 40 amostras do outro caso e, assim, podem estar localizadas em posições mais informativas, o que certamente ocorreu por acaso.

Embora as observações mencionadas nesta seção sejam específicas para a tarefa de classificação considerada, a metodologia que foi seguida na atribuição do desempenho do HE-SVMs e a variabilidade de configuração de acordo com aspectos diferentes do conjunto de dados de treinamento é extensível a outros experimentos de classificação.

5.4.3.2 Reconhecimento de padrões de expressão gênica

Este experimento diz respeito à tarefa de identificação de células cancerígenas a partir de dados de expressão gênica. Experimentos baseados em microarray de DNA estão sendo utilizados para coletar informações de amostras de tecidos. O problema de classificação

examinado nesta seção foi inicialmente investigado por ALON *et al.* (1999). Este problema de classificação representa um grande desafio, uma vez que o número de atributos por padrão é muito alto e as amostras de treinamento são esparsas. GUYON *et al*. (2002) exploraram a aplicabilidade de SVMs para este problema. No entanto sua análise foi centrada principalmente nas vantagens obtidas quando da aplicação de um algoritmo de eliminação recursiva de atributos, antes da aplicação da abordagem SVM.

A idéia é discriminar entre dois grupos de pacientes, pacientes doentes (classe 1) e pacientes normais (classe 2), baseado nos dados de expressão gênica do tecido do cólon. A partição é 40 amostras para a primeira classe e 22 amostras para a segunda classe. Uma vez que não há conjunto de treinamento e teste definidos a priori, foram utilizadas 31 amostras para treinamento, 16 amostras para seleção dos componentes e 15 amostras para teste, todas elas escolhidas randomicamente.

Dois tipos de experimentos foram realizados, com 15 replicatas respeitando a mesma porcentagem de amostras para cada conjunto, mas com atribuição aleatória. Em outras palavras, o conjunto de treinamento tem sempre o mesmo número de amostras, em cada replicata, mas seu conteúdo pode ser bem diferente do conteúdo dos demais conjuntos de treinamento, associados às outras replicatas. No primeiro experimento, foram realizadas simulações em que os parâmetros do *kernel* para todos os componentes foram fixados com base nos valores padrões apresentados no Capítulo 4. A Tabela 5.5 apresenta a porcentagem[3] do número médio de padrões classificados incorretamente para cada componente SVM em relação ao HE-SVMs, com o respectivo desvio padrão (SD) para as 15 replicatas. Um valor percentual alto indica um desempenho relativo ruim para um dado SVM. Os valores em negrito na Tabela 5.5 indicam o componente que obteve o melhor desempenho médio em relação ao HE-SVMs.

Os dados de entrada foram normalizados de acordo com o intervalo de definição de cada *kernel* (veja Tabela 5.6). Os resultados apresentados na Tabela 5.5 correspondem a

[3] Seja % P a porcentagem média de padrões classificados incorretamente pelo SVM com *kernel* (i), n_i e n_H o número médio de padrões classificados incorretamente pelo SVM com *kernel* (i) e pelo HE-SVMs, respectivamente, logo P_i pode ser escrito como:

$$\%P = \frac{(n_i - n_H)}{n_H}$$

C = 100 e C = 1000, embora simulações para outros valores foram realizadas, sem produzir alterações significativas no desempenho. Pode ser observado que diferentes valores do parâmetro C podem conduzir a comportamento distintos para cada *kernel*. Por exemplo, o *kernel* tangente hiperbólica se beneficiou drasticamente com o aumento de C, ao passo que o efeito oposto ocorreu com o *kernel* polinomial.

	C = 100				C = 10000			
	Conjunto de seleção		Conjunto de teste		Conjunto de seleção		Conjunto de teste	
	% P	S.D.	*% P*	S.D.	*% P*	S.D.	*% P*	S.D.
Linear	**16,279**	**1,291**	5,556	1,246	21,951	1,291	15,152	1,246
Polinomial	**16,279**	**1,291**	5,556	1,246	21,951	1,291	15,152	1,246
RBF	46,512	1,424	27,778	1,335	53,659	1,424	39,394	1,335
ERBF	25,581	1,242	**5,556**	**1,060**	31,707	1,242	15,152	1,060
Tangente	109,302	0,00	108,333	0,00	**7,317**	**1,387**	**3,030**	**0,961**
Fourier	248,837	0,00	316,667	0,00	265,854	0,00	354,545	0,00
Splines	44,186	1,598	11,111	1,589	51,220	1,598	21,212	1,589
Bn-splines	248,837	0,00	316,667	0,00	265,854	0,00	354,545	0,00
HE-SVMs	0,0	1,302	0,0	1,056	0,0	1,335	0,0	1,014

Tabela 5.5 – Resultados comparativos entre o HE-SVMs e SVMs isolados. Note que aqui os parâmetros dos *kernels* não foram ajustados por validação cruzada.

Tipo de *Kernel*	Intervalo De Normalização	Parâmetros do *kernel* Validação Cruzada	
		Intervalo	Passo
Linear	[-1; 1]	------	------
Polinomial	[-0,5 ; 0,5]	$d \in$ [2 ; 8]	1
RBF	[-1 ; 1]	$\sigma \in$ [0,1 ; 3]	0,75
ERBF	[-1 ; 1]	$\sigma \in$ [0,1; 3]	0,75
Tangente	[-0,8 ; 0,8]	$v \in$ [-1 ; 1] , c = 0,01	0,3
Fourier	[-π/2 ; π/2]	$l \in$ [0 ; 2]	1
Splines	[0 ; 0,8]	------	------
Bn-splines	[-1 ; 1]	$n \in$ [0 ; 1]	1

Tabela 5.6 – Intervalos de normalização e intervalo de busca dos parâmetros do *kernel*

A fim de melhorar o desempenho dos componentes, foi realizado um processo de seleção dos parâmetros de cada *kernel*. Tal procedimento foi realizado variando o valor dos parâmetros do *kernel* associado, de acordo com o intervalo e os passos fornecidos na Tabela 5.6, que foram fixados arbitrariamente. Poderia ter sido seguida, de forma alternativa, à abordagem proposta por CRISTIANINI *et al.* (1999), que iterativamente vai incrementando o valor do parâmetro do *kernel* até que o desempenho comece a piorar. Para cada tipo de *kernel* e para cada valor do parâmetro do *kernel*, um processo de validação cruzada com 10 partições sobre o conjunto de dados de treinamento (N_{tr}=31) foi realizado. A Tabela 5.7 mostra o desempenho comparativo entre a configuração resultante para cada *kernel* em relação ao HE-SVMs. Pode ser observado que o desempenho relativo daqueles *kernels* que não passaram por um processo de otimização (linear e Spline) diminui. Isto se deve ao fato de que o ensemble formado é agora composto de componentes otimizados. Como no experimento anterior, o comportamento dos *kernels* continua muito dependente do valor escolhido para o parâmetro *C*.

	C = 100				*C* = 10000			
	Conjunto de seleção		Conjunto de teste		Conjunto de seleção		Conjunto de teste	
	% Média	S.D.	% Média	S.D.	% Média	S.D.	% Média	S.D.
Linear	25	1,291	26,666	1,246	42,857	1,291	46,154	1,246
Polynomial	25	1,291	26,666	1,246	57,143	1,759	80,769	2,232
RBF	32,5	1,407	16,666	1,397	54,286	1,639	50,000	1,502
ERBF	**15**	**1,534**	**6,666**	**1,246**	28,571	1,512	23,077	1,246
H. Tangent	155	1,656	200	2,070	**8,571**	**1,302**	**3,846**	**1,014**
Fourier	125	0,00	150	0,00	157,143	0,00	188,462	0,00
L. Splines	55	1,598	33,333	1,589	77,143	1,598	53,846	1,589
Bn-splines	125	0,00	150	0,00	157,143	0,00	188,462	0,00
HE-SVM	0,0	1,345	0,0	1,134	0,0	1,175	0,0	1,163

Tabela 5.7 – Comparação entre o HE-SVMs e diferentes SVMs isoladas, com parâmetros do *kernel* ajustados via validação cruzada

Tipo de *Kernel*	Sem Validação Cruzada		Com Validação Cruzada	
	C = 100	*C* = 10000	*C* =100	*C* = 10000
Linear	0	0	0	0
Polinomial	9	3	4	2
RBF	1	0	6	3
ERBF	6	5	6	6
Tangente	0	8	0	4
Fourier	0	0	0	0
Splines	0	0	0	0
Bn-splines	0	0	0	0
HE-SVM	1,067	1,067	1,067	1,00

Tabela 5.8 – Freqüência dos *kernels* na configuração final do HE-SVMs

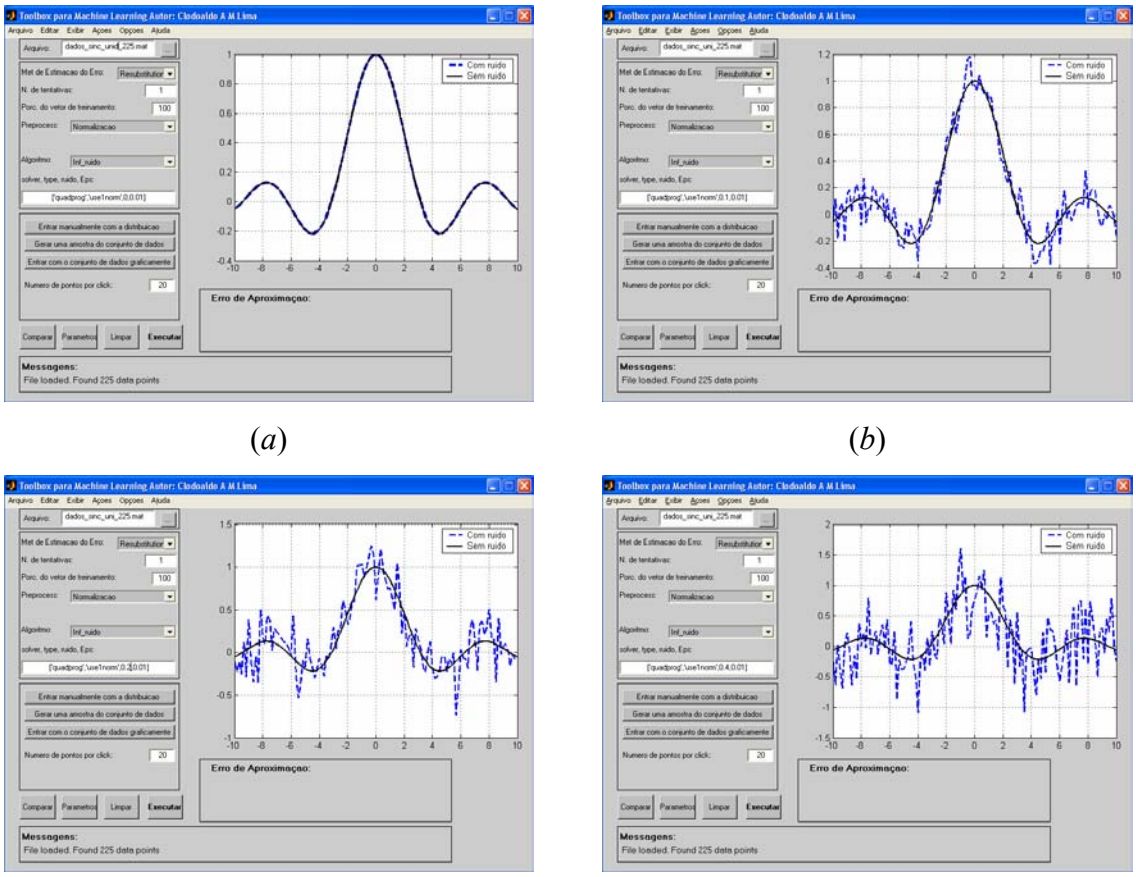
A Tabela 5.8 apresenta a freqüência de cada *kernel* na configuração final de todos os HE-SVM produzidos nas 15 replicatas, levando em consideração os experimentos com e sem validação cruzada. A última linha da Tabela 5.8 mostra o tamanho médio (número de componentes) do HE-SVM resultante. Devido ao tamanho limitado do conjunto de seleção ($N_s = 16$), observa-se uma pequena diferença de desempenho sobre o conjunto de dados de seleção entre as SVMs. Isto justifica o fato do ensemble ser formado por poucos componentes, pois não há diversidade a ser explorada.

5.4.4 Ensembles e conjunto de dados ruidosos

Um dos resultados mais importantes associados com o HE-SVMs é a possibilidade de abrandar a influência do nível de ruído junto aos dados de treinamento, uma vez que o desempenho de cada *kernel* é fortemente influenciado pelo nível de ruído.

Os resultados a serem apresentados nesta seção serão principalmente devotados a indicar que um ensemble é capaz de superar o desempenho da melhor SVM (S-SVM), para tarefas específicas de regressão e classificação. O objetivo desta seção é investigar aspectos de sensibilidade a ruído do HE-SVMs, quando comparado com a melhor SVM isolada.

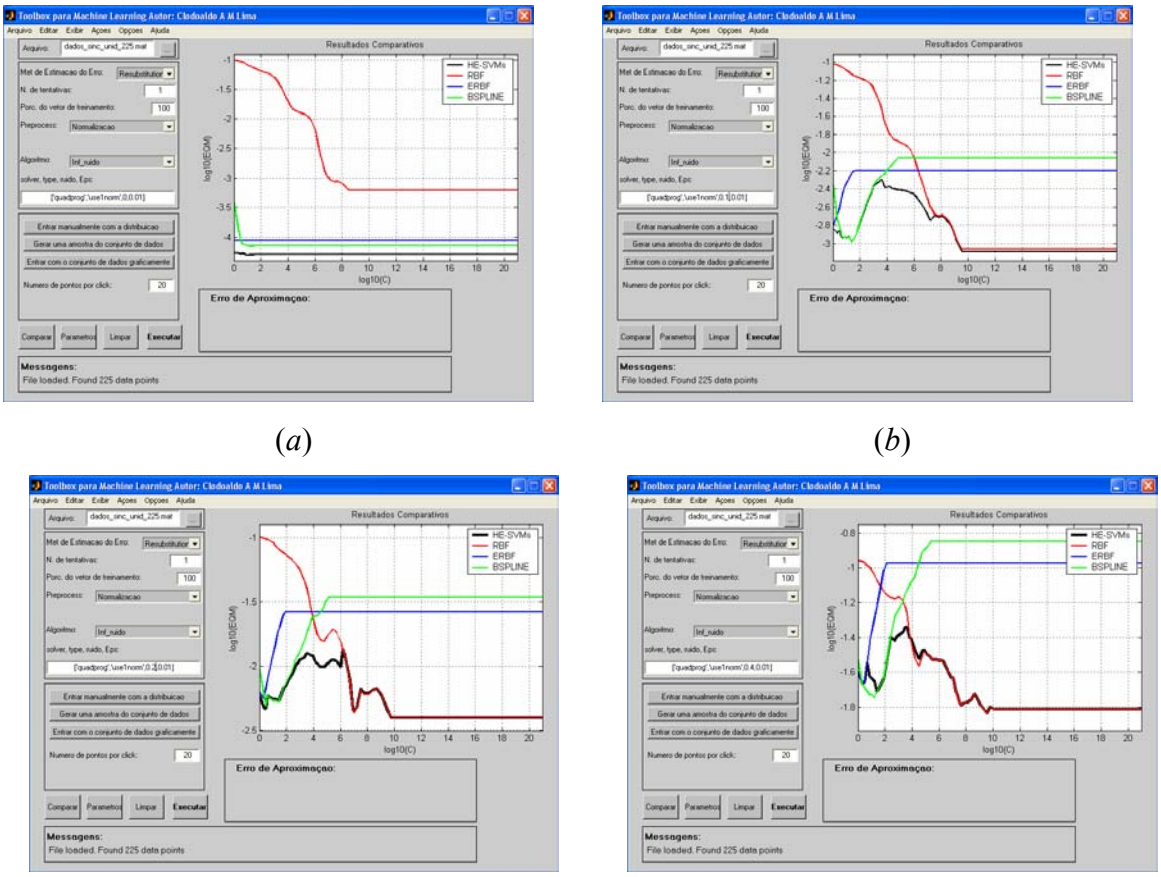
Considerando novamente a função $g^{(1)}$ (veja Apêndice A) com diferentes níveis de ruído representados na Figura 5.10, a evolução do desempenho dos três melhores SVMs (aquele empregando B-spline, ERBF, e RBF como função *kernel*), comparada com o desempenho do HE-SVM, é apresentada na Figura 5.10, quando o parâmetro C é variado no intervalo [0,0067, 4,8517e+008], e ε é fixado em 0,1. O ruído escolhido é gaussiano e tem média zero e desvio padrão igual a 0,1, 0,2 e 0,4. Dois aspectos relevantes devem ser mencionados: (*i*) a notável variação de desempenho de um *kernel* em função de C, na presença de ruído; e (*ii*) a manutenção do modelo HE-SVMs como melhor modelo para qualquer que seja o valor de C, exceto para um ruído com desvio padrão 0,4, onde o HE-SVMs teve desempenho pior que o melhor SVM para alguns valores de C.



(*a*) (*b*)

(*c*) (*d*)

Figura 5.10 – Função $g^{(1)}$ com diferentes níveis de ruído: (*a*) sem ruído; (*b*) ruído com σ = 0,1; (*c*) ruído com σ = 0,2; (*d*) ruído com σ = 0,4.



(*a*) (*b*)

(*c*) (*d*)

Figura 5.11 – Evolução do desempenho da abordagem HE-SVMs e SVMs isolados para a variação no parâmetro *C*, com conjunto de dados de treinamento sujeito a diferentes níveis de ruído: (a) sem ruído; (b) ruído com desvio padrão igual a 0,1; (c) ruído com desvio padrão igual a 0,2; (d) ruído com desvio padrão igual a 0,4; $\varepsilon = 0{,}01$.

5.5 Seleção automática de componentes para HE-SVMs

A etapa de seleção de modelos está associada à imposição de restrições sobre o algoritmo de aprendizado, com o propósito de minimizar o erro de generalização da máquina de aprendizado, ou seja, minimizar a complexidade intrínseca do modelo de modo a atender adequadamente os requisitos da aplicação (KEARNS *et al.*, 1997). Estes requisitos da aplicação, diretamente vinculados à complexidade ótima do modelo, devem ser

extraídos dos dados amostrados, de alguma forma. Para tanto, normalmente são reservadas parte das amostras disponíveis para compor o chamado conjunto de dados de seleção.

Sob a perspectiva do HE-SVMs, o processo de seleção de modelos consiste em escolher o subconjunto de máquinas de vetores-suporte, dentre as candidatas disponíveis, que irão compor o ensemble, visando maximizar a capacidade de generalização do ensemble. Tais idéias estão de acordo com os argumentos apresentados por ZHOU *et al.* (2002) para justificar a escolha de um subconjunto de redes neurais candidatas, em lugar de considerar todas as redes neurais disponíveis na composição do ensemble.

Do ponto de vista do HE-SVMs, o problema de seleção de modelos pode ser formulado como segue: Dada uma população $P = \{1, 2,..., M\}$ de máquinas de vetores-suporte heterogêneas, ou seja, que empregam *kernels* distintos, treinadas independentemente, escolher um subconjunto P^* dentre todos os $2^M - 1$ subconjuntos possíveis de tal modo que o ensemble correspondente produza o melhor desempenho.

Neste sentido, a seleção ótima de componentes é dependente do critério de desempenho adotado para medir a capacidade de generalização individual dos componentes e do ensemble resultante de cada subconjunto de SVMs candidatas. Sendo assim, justifica-se uma investigação mais aprofundada de diferentes critérios de seleção de modelos.

5.5.1 Seleção de modelo baseado no critério de Perrone e Cooper (PSel)

Conforme mencionado no Capítulo 2, em um ensemble, quando se aumenta o tamanho da população de componentes P, diminui acentuadamente a probabilidade de que os erros cometidos pelos componentes sejam mutuamente independentes. Na ausência de divergência em generalização, a adição de novos componentes não produz ganho de desempenho do ensemble, representando assim um desperdício de recursos computacionais. Além disso, pode haver perda de desempenho para o ensemble, o que inclusive sustenta a proposição de métodos de poda de componentes, conforme será visto mais adiante).

Assim, é interessante buscar o menor subconjunto de componentes que maximize a capacidade de generalização do ensemble correspondente. Uma possibilidade seria examinar todos os $2^M - 1$ subconjuntos, mas para M muito grande esta busca torna-se impraticável computacionalmente. Para tornar a busca tratável, PERRONE & COOPER (1993) propuseram dispor os elementos da população em ordem crescente do valor do EQM e então gerar um conjunto de $l \leq M$ componentes pela adição sucessiva de elementos ordenados de P, mas descartando aqueles elementos cuja adição não conduz a ganho de desempenho. Em cada tentativa de adição de um componente ao ensemble, verifica-se se há ganho de desempenho em relação ao ensemble anterior. Em caso positivo, o componente é mantido; caso contrário, é descartado. Esta abordagem é conhecida como um *método construtivo*, uma vez que se inicia com uma quantidade pequena de componentes (normalmente apenas com o componente de melhor desempenho, quando atuando isolado dos demais) e busca-se incorporar novos componentes quando isto representar um ganho de desempenho. Uma expressão analítica que pode ser empregada na decisão de inclusão ou não de candidatos foi apresentada no Capítulo 2 (equações 2.17 e 2.23, para regressão e classificação, respectivamente).

Repare que a proposta de PERRONE & COOPER (1993) representa um algoritmo aproximado, pois apenas um número reduzido de subconjuntos são avaliados, dentre todas as $2^M - 1$ possibilidades. Se um determinado candidato a compor o ensemble não apresenta contribuição significativa para o ganho numa determinada etapa de construção do ensemble, ele é definitivamente descartado. No entanto, para outras configurações de ensemble, que não aquela considerada naquele momento, é possível que aquele mesmo candidato se mostre em condições de contribuir para o desempenho do ensemble. Porém, esta possibilidade nunca será avaliada.

5.5.2 Seleção de modelo baseado no critério de Zhou (ZSel)

Ao contrário do critério de seleção anterior, ZHOU *et al.* (2002) propôs gerar inicialmente um ensemble composto de todos os elementos de P e, posteriormente, excluir alguns componentes, visando aumentar o desempenho global do ensemble. ZHOU *et al.*

(2002) não analisou a questão da ordenação dos elementos de *P*. No entanto, tal ordenação é importante para que o problema possa ser tratado computacionalmente. Um dos critérios a serem utilizados nesta tese será a ordenação via valor decrescente do EQM. Este critério pode ser considerado como uma estratégia gulosa, ou seja, um algoritmo aproximado, uma vez que tenta eliminar inicialmente os elementos de *P* com pior desempenho individual. Esta abordagem é conhecida como *método de poda*. Uma expressão analítica que pode ser empregada na decisão de exclusão ou não de candidatos foi apresentada no Capítulo 2 (equações 2.14 e 2.28, para regressão e classificação, respectivamente).

5.5.3 Critério de ordenação baseado na dimensão VC (VCSel)

A dimensão VC (*h* – veja Capítulo 4 para maiores detalhes) fornece uma medida alternativa, de acordo com a teoria do aprendizado estatístico, para atribuir a capacidade de generalização de uma SVM. Tal abordagem tem a vantagem de não necessitar da escolha a priori de um conjunto de dados de seleção para ordenar os componentes, além de fornecer uma medida de desempenho cuja confiabilidade não é diretamente influenciada pela quantidade de amostras disponíveis, como no caso do EQM. A utilização da dimensão VC como critério de ordenação para o HE-SVMs está de acordo com a metodologia de SVM, uma vez que este implica na minimização do risco estrutural (LIMA *et al.*, 2002). Um caso específico vinculado à dimensão VC seria a aplicação da margem de separação como critério de ordenação. Isto pode ser realizado desde que o raio da esfera contendo todas as amostras no espaço de alta dimensionalidade seja limitado, conforme discutido no Capítulo 4, seção 4.8.

5.5.4 Seleção de modelos via algoritmo genético (GASel)

Um algoritmo genético (BÄCK *et al.*, 1997) constitui uma técnica de otimização global que tem mostrado bom desempenho junto a problemas de otimização combinatória e otimização multimodal. Estes algoritmos vinculados a técnicas de computação evolutiva não garantem a obtenção da solução ótima, mas tendem a produzir boas soluções pela

avaliação de um número reduzido de soluções candidatas do espaço de busca (FOGEL, 1999).

Sendo assim, a seleção de modelos, por envolver um espaço de soluções candidatas de cardinalidade $2^M - 1$ (implicando assim em uma explosão combinatória de candidatos, conforme M cresce) e por envolver multimodalidade (visto que a importância relativa de cada componente é variável para cada composição do ensemble), pode ser realizada via algoritmos genéticos com codificação binária. Em tal contexto, ZHOU *et al.* (2002) aplicaram uma metodologia baseada em um algoritmo genético para buscar uma seleção ótima de componentes, tendo redes neurais artificiais como candidatos. A combinação resultante apresentou desempenho de generalização superior àquelas produzidas através de métodos tradicionais para ensemble, tais como *boosting* e *bagging*.

No caso do GASel, os cromossomos são codificados como vetores binários de comprimento igual a M, no qual cada bit indica a presença ('1') ou ausência ('0') de um dos elementos de P no ensemble. A população inicial é gerada randomicamente (via uma distribuição uniforme) e as gerações sucessivas são geradas por meio de operadores genéticos (*crossover* de um ponto e mutação simples), aplicados nos indivíduos selecionados. A seleção elitista, onde 10% dos melhores indivíduos são mantidos na população, é empregada juntamente com o operados de seleção de amostragem estocástica universal (BÄCK *et al.*, 1997). Os indivíduos são selecionados de acordo com a regra de atribuição de adaptabilidade, conhecida como função de *fitness*, dada por:

$$Fitness_k = \beta \cdot EQM_k + (1 - \beta) \cdot D,$$

onde o primeiro termo do lado direito da expressão acima refere-se ao desempenho do ensemble correspondente ao k-ésimo vetor binário, expresso na forma de uma medida de erro quadrático médio sobre o conjunto de dados de seleção. Já o segundo termo leva em conta quão diversos são os componentes, sendo que RUTA & GRABYS (2001) apresentaram uma proposta para D no caso de problemas de classificação e LIU & YAO (1999) propuseram uma medida de correlação negativa para problemas de regressão.

O parâmetro β deverá ser ajustado de acordo com o problema. Ele fornece uma medida de compromisso entre as duas parcelas mencionadas. Além disso, para promover a parcimônia no projeto do ensemble final, foi adotado o princípio do comprimento da

descrição mínima (MDL – *minimum description length*) (KEARNS *et al.*, 1997) na seleção do modelo: quando o desempenho de duas configurações de HE-SVMs é muito semelhante, um termo de penalização proporcional ao número de componentes selecionados é introduzido.

5.5.5 Experimentos

Seis tipos de ensemble foram gerados neste experimento, cada um com um método de seleção de componentes. Os *kernels* adotados foram os mesmos dos experimentos da seção 5.4.2. Ao contrário dos experimentos anteriores, nos quais foi utilizada somente a função de perda ε-insensível, nesta seção serão aplicadas todas as três funções de perda apresentadas no Capítulo 4. Logo, haverá 24 modelos de SVM com chance de serem selecionados e combinados para produzir P^*, isto é, o tamanho do espaço de busca é $O(2^{24})$. Nos experimentos com GASel, foram adotados os seguintes parâmetros, após várias tentativas de sintonia: uma população com 100 indivíduos, 100 gerações, a probabilidade de *crossover* igual a 0,7 e uma probabilidade de mutação de 0,07. Para todos os métodos, os resultados apresentados correspondem a uma média de 10 replicatas.

Os conjuntos de dados tratados foram os seguintes: problema de diagnóstico de câncer de seio (*breast cancer diagnosis*) (BCD) e o problema de hepatite (HEP), ambos disponíveis na base de dados do UCI (BLAKE & MERZ, 1998) e o problema de reconhecimento de câncer via dados de expressão gênica, investigado originalmente por ALON *et al.* (1999).

O problema BCD possui 699 amostras, onde 458 são indicadas como câncer benigno (65,5%) e 241 são indicadas como maligno (34,5%). Nos experimentos, foram usadas as mesmas partições do trabalho de LEE *et al*. (2001), cujo conjunto de treinamento, seleção e teste eram compostos de 349, 175 e 175 amostras, respectivamente.

O conjunto de dados de hepatite está associado a um problema de classificação binária com 155 exemplos, sendo que a tarefa do classificador é indicar, de acordo com o perfil do paciente, se ele corre ou não risco de morte. Há 19 atributos de entrada e 167 valores faltantes, algo que aumenta a dificuldade do problema. A distribuição dos dados

32/123 é mantida em todo o particionamento do conjunto de dados. Os conjuntos de treinamento, seleção e teste foram compostos de 78, 39 e 38 padrões, respectivamente.

Por sua vez, o conjunto de reconhecimento de câncer via dados de expressão gênica já foi analisado na seção 5.4.2. A partição utilizada aqui será a mesma já adotada, ou seja, 31 amostras para treinamento, 16 para seleção dos componentes e 15 para teste.

Os resultados para diferentes métodos de combinação são apresentados na Tabela 5.9. Analisando-a, é possível observar que a abordagem GASel produziu, em média, melhores resultados, comparados com os outros métodos de seleção. O tempo computacional gasto para obtenção da solução via GASel não foi da mesma ordem que aqueles obtidos por qualquer outro método de seleção, uma vez que, para aplicação de todos os métodos, é necessário a obtenção das SVMs e suas respectivas saídas para o conjunto de seleção. Além disso, GASel tem a vantagem de indicar múltiplas configurações de componentes com desempenho semelhante junto ao conjunto de dados de seleção, constantes da população na geração final, algo que pode ser explorado na indicação experimental de quais tipos de *kernel* são os mais indicados na construção de uma máquina de aprendizado baseado em SVM, para uma determinada tarefa. O desempenho informado na Tabela 5.9 para todos os métodos de seleção é bastante significativo e é comparável aos melhores já apresentados na literatura (LEE *et al.*, 2001; GUYON *et al.*, 2002).

Método seleção	Critério de Ordenação	BCD		HEP		Alon	
		Média	SD	Media	SD	Média	SD
Construtivo	EQM	96,38	1,19	82,11	5,06	82,67	3,65
	VC	96,57	0,99	80,00	4,40	82,67	3,65
	Margem	96,57	0,99	81,58	4,56	82,67	5,96
Poda	EQM	96,70	0,90	82,11	4,90	83,15	3,42
	VC	96,81	1,01	81,38	4,22	82,67	3,65
	Margem	96,70	1,21	81,13	3,98	82,67	4,00
GASel, $\beta = 0{,}5$	-----	**97,14**	0,57	**82,63**	3,00	**84,00**	3,65

Tabela 5.9 – Resultado para diferentes métodos de seleção de componentes

5.6 Ensemble com múltiplos estágios para SVM

Conforme constatado nos experimentos anteriores, a combinação das saídas de componentes diferentes pode conduzir a um melhor desempenho de generalização.

Estratégias de combinação tradicionais para ensemble incluem média simples (TUMER & GHOSH, 1995), média ponderada (JACOBS, 1995) e voto majoritário (HANSEN & SALAMON, 1990), este último no caso de classificação. No entanto, não há um procedimento sistemático de escolha de qual método de combinação utilizar. A escolha depende principalmente da natureza da aplicação do ensemble, das dimensões envolvidas e da qualidade dos dados de treinamento, ou dos erros gerados sobre diferentes regiões do espaço de entrada. Um método de combinação pode se dar muito bem em um dado problema de regressão ou classificação, mas pode apresentar queda acentuada de desempenho para outros problemas, por exemplo, devido a uma baixa representatividade do conjunto de dados de seleção. Além disso, componentes diferentes poderão influenciar na escolha do método de combinação mais apropriado. Atualmente, há muitos experimentos empíricos publicados na literatura, mas nenhum deles propõe um critério de escolha ótimo para o método de combinação a ser utilizado. Mais resultados teóricos e experimentais são necessários para se adotar um procedimento sistemático.

5.6.1 Trabalhos relacionados

Em 1992, WOLPERT propôs uma técnica de combinação de classificadores chamada empilhamento (do inglês *stacking*). Empilhamento atua sobre duas áreas importantes na construção de ensembles: a preparação dos dados para geração dos componentes e a combinação destes. Primeiramente, tal abordagem usa a idéia de re-amostragem sem reposição para selecionar dados de treinamento a serem utilizados pelos componentes. Segundo, ela utiliza um componente (chamado *segundo nível*) para combinar as saídas dos componentes do nível inferior (chamado *primeiro nível*). Uma característica da generalização empilhada (do inglês *stacked generalization*) é que a informação fornecida para o primeiro nível de componentes vem de múltiplos particionamentos do conjunto de dados original. Assim, o conjunto de treinamento é dividido em duas partes: uma parte para treinamento dos componentes do primeiro nível (via re-amostragem) e a outra parte para treinamento do segundo nível. Após o treinamento dos componentes do primeiro nível, as saídas destes componentes para o segundo conjunto de dados de treinamento são

calculadas. Estas saídas junto ao segundo conjunto de treinamento são utilizadas como entrada para treinamento dos componentes do segundo nível. Uma das desvantagens desta estratégia é a necessidade de uma grande quantidade de dados para treinamento dos componentes.

Inspirado na estratégia acima, alguns pesquisadores perceberam que é possível construir um novo método de combinação usando idéia similar. Por exemplo, LEE & SRIHARI (1993) propuseram uma rede neural de uma camada escondida para combinar as saída de classificadores para um problema de reconhecimento de digito numérico. PATRIDGE & GRIFFIH (1995) apresentaram uma abordagem chamada *selector-net*. O *selector-net* era também uma rede neural que combinava a saída de outras redes neurais. ZENG *et al.* (2000) usou uma única rede neural para aproximar o método de combinação baseado em votação.

YANG *et al.* (2002) estenderam a idéia de empilhamento e propuseram uma abordagem chamada *ensemble de redes neurais com múltiplos estágios*, tendo uma rede neural de uma camada escondida como um combinador. Todas esta abordagens diferem principalmente na forma de geração dos componentes e na forma de treinamento do combinador.

Entretanto, tais esquemas de combinação têm duas grandes limitações. Primeiramente, não há um critério de seleção para filtrar as entradas para o combinador, ou seja, selecionar quais redes formarão parte do ensemble. Segundo, o combinador utilizado geralmente é uma rede neural que é sensível a um mínimo local e que, com o aumento do número de componentes, há um aumento da dimensão da entrada para o combinador, conseqüentemente aumentando a complexidade deste.

Para atuar justamente nestas limitações, é proposto um modelo de SVM como um combinador. A motivação para tal abordagem está baseada no alto poder de generalização, na otimalidade global e na independência da dimensão dos dados de entrada. Portanto, a idéia do chamado *ensemble com múltiplos estágios para SVM* (daqui por diante, EME-SVM) é produzir combinadores menos sensíveis aos problemas citados acima e capazes de implementar um mapeamento não-linear tomando as saídas de todos os componentes e também os dados de entrada como suas entradas. A abordagem em múltiplos estágios

apresenta duas extensões frente a um combinador linear: a combinação é agora uma função dos dados de entrada e pode ser não-linear. Esta abordagem difere da abordagem hierárquica de SVM proposta por KWOK (1998) (a qual consiste de um SVM na segunda camada, onde a saída dos vários SVMs na camada inferior correspondem à entrada do SVM na camada superior) sob dois aspectos. O primeiro aspecto diz respeito à forma de treinamento: enquanto KWOK (1998) propõe um único problema QP para treinamento de todos os componentes, inclusive do combinador, aumentando drasticamente a dimensionalidade da matriz de *kernels* e tornando a solução bastante difícil de ser obtida, nesta formulação tanto os componentes quanto o combinador são treinados individualmente. O outro aspecto diz respeito ao conjunto de entrada para o combinador: enquanto KWOK (1998) propõe como entradas somente as saídas dos componentes, nesta formulação os dados de entrada são agrupados com as saídas dos componentes, com o objetivo de fornecer mais informação para o combinador.

5.6.2 Descrição da abordagem EME-SVM

Um ensemble com múltiplos estágios para SVM é uma união de várias SVMs com diferentes tipos de *kernel*, cujas decisões individuais juntamente com os dados de entrada são combinadas por outro SVM para classificar ou estimar os exemplos de teste, conforme Figura 5.12. A saída $f(\mathbf{x})$ é dada por:

$$f(\mathbf{x}) = \mathbf{w}_c^T \varphi_c(\mathbf{z}) + b_c, \tag{5.8}$$

com $\mathbf{z} = [\mathbf{w}_1^T \varphi_1(\mathbf{x}) + b_1 \quad \ldots \quad \mathbf{w}_M^T \varphi_M(\mathbf{x}) + b_M \quad \mathbf{x}]$, onde $\mathbf{z}$, $\mathbf{w}_c$ e b_c são, respectivamente, o vetor de saídas dos componentes, os pesos e o bias do combinador.

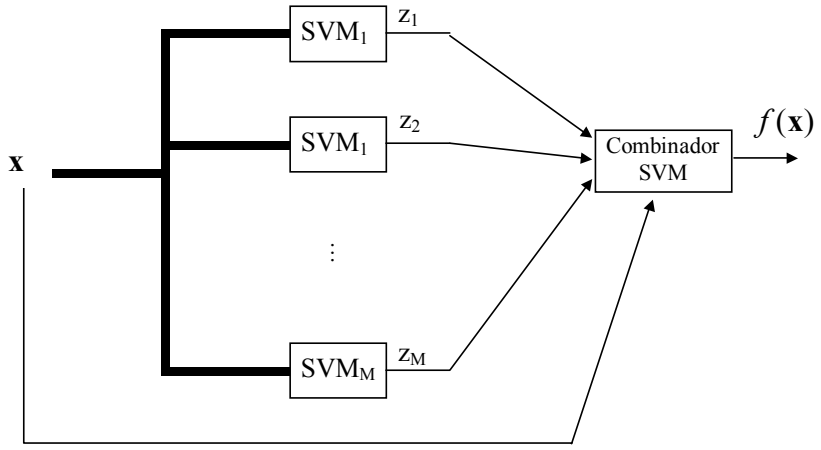


Figura 5.12 – Arquitetura de um EME-SVM

Vantagens do ensemble com múltiplos estágios para SVM incluem:

- EME-SVM pode ser aplicado tanto para problemas de classificação como para problemas de regressão;
- Para cada componente, métodos de preparação dos dados podem ser aplicados;
- Os componentes podem ser treinados individualmente usando diferentes tipos de *kernel*;
- A capacidade de generalização dos componentes pode ser melhor combinada;
- Maior flexibilidade no mapeamento de combinação, uma vez que a saída do ensemble é uma função da entrada e das saídas dos componentes.

5.6.3 Treinamento da abordagem EME-SVM

Toma-se um conjunto de dados de treinamento e particiona-se o mesmo em quatro subconjuntos, a saber: dois conjuntos de dados de treinamento (sendo um conjunto para treinamento dos componentes e outro para treinamento do combinador, cada um com 30% dos dados disponíveis), um conjunto de seleção e outro de teste, cada um com 20% dos dados disponíveis. Os dados de treinamento dos componentes podem ser pré-processados

por vários métodos, por exemplo, re-amostragem adaptativa, a fim de gerar comportamentos diversos para os componentes baseados em modelos SVM. Após o treinamento dos componentes, um dos métodos de seleção propostos anteriormente é utilizado para selecionar os melhores componentes, baseados no conjunto de seleção. Selecionados os componentes, a saída destes para o segundo conjunto de treinamento é calculada. Logo, o combinador baseado em SVM é treinado tendo como entrada as saídas geradas pelos componentes, concatenadas com o respectivo conjunto de treinamento. Após treinado o combinador, todo o sistema é testado com base no conjunto de teste. A seguir, é apresentado um pseudo-código para treinamento de um EME-SVM.

1 Fase de geração (para os componentes candidatos)
  1.1 Para cada candidato *j*
  1.2 Escolha um *kernel*
  1.3 Escolha um parâmetro *C*
  1.4 Aplique o procedimento de treinamento para o candidato *j*
2 Fase de seleção
  2.1 Obtenha o desempenho dos candidatos com um conjunto de seleção
  2.2 Disponha em ordem crescente os candidatos de acordo com o seu desempenho
  2.3 Aplique um dos critérios de seleção
3 Fase de treinamento do combinador
  3.1 Calcule a saída dos candidatos selecionados, usando o segundo conjunto de treinamento
  3.2 Concatene estas saídas com as entradas do conjunto de dados
  3.3 Treine o combinador com este conjunto de dados concatenados
4 Obtenha o desempenho do EME-SVM usando o conjunto de teste.

Algoritmo 5.1 – Algoritmo para treinamento da abordagem EME-SVM

5.6.4 Experimentos

Para investigar o desempenho do EME-SVM, quatro problemas foram obtidos da base de dados UCI de aprendizado de máquina (BLAKE & MERZ, 1998), sendo que três deles correspondem a problemas de classificação e um a problema de regressão.

Os conjuntos de dados utilizados para classificação estão associados a diagnóstico de câncer do cólon (BCD) (já mencionado na seção 5.5.5), diabetes (DIAB), com 768 amostras e 8 atributos, e diagnóstico de câncer Wisconsin (WDBC), com 569 amostras e 30 atributos. Todos estes conjuntos correspondem a problemas de classificação binária.

O conjunto de dados para regressão consiste do problema de *Housing Boston*, que contém informação coletada pelo serviço de censo dos Estados Unidos a respeito das habitações na área popular de Boston. Este contém 506 amostras, 13 atributos e duas saídas a serem estimadas: **nox**, que é o nível de óxido nitroso; **price** que é o valor médio das casas. Cada saída foi tratada independentemente, isto é, um ensemble para cada uma.

Para o problema de classificação, além dos 24 modelos de SVM (que correspondem aos oito tipos de *kernel* e às três funções de perda definidas no Capítulo 4), outras formulações para o SVM foram investigadas, como LS-SVM (apresentado no Capítulo 4). Uma descrição detalhada das diferentes formulações utilizadas pode ser encontrada em LIMA *et al.* (2004). O parâmetro C para todas as formulações de SVM foi fixado em 10^4. Além disso, a partir de cada componente (escolhido de uma das formulações, de um dos *kernels* e de uma das funções de perda), foram gerados 11 novos componentes, sendo cinco gerados via *bagging*, cinco via *boosting* e um componente incluindo todo o conjunto de treinamento. Subseqüentemente, durante o processo de seleção, dois conjuntos de componentes foram obtidos, um pelo método de poda (EME-SVMp) e o outro pelo método construtivo (EME-SVMc). Depois de selecionados os componentes do ensemble (veja Algoritmo 5.1), o segundo conjunto de treinamento é apresentado para os componentes e usado como dados de treinamento para o combinador. Depois do processo de treinamento, um conjunto de teste é apresentado ao ensemble resultante para medir o desempenho. Como um cenário comparativo, variações de SVMs tomadas isoladamente foram aplicadas para o mesmo problema de classificação, com busca exaustiva para os parâmetros da função *kernel* e para o parâmetro C. A partição adotada aqui é a mesma considerada acima, mas agora o subconjunto de treinamento contém 60% dos dados originais (não há combinador para ser treinado). A Tabela 5.10 apresenta a média do desempenho de cada abordagem considerando 10 execuções.

Abordagem	Taxa média de classificação correta		
	BCD	WDBC	Diabetes
SVM	97,086	95,660	74,510
LS-SVM	97,102	96,775	74,430
Bagging-SVM	97,122	97,345	77,163
Boosting-SVM	97,102	97,315	77,743
EME-SVMp	97,222	97,370	78,014
EME-SVMc	97,127	97,345	78,342

Tabela 5.10 – Resultado para os conjuntos de dados de classificação

Para o problema de regressão, foram gerados apenas 32 possíveis componentes, sendo 24 via modelo tradicional de SVM e 8 via modelo LS-SVM (que corresponde aos oito tipos de *kernel*, uma vez que para esta abordagem existe somente uma função de perda possível). O processo de treinamento do modelo EME-SVM (neste caso, não houve processo de re-amostragem) é o mesmo mencionado acima. Para os modelos individuais de SVM e LS-SVM, foi realizada busca exaustiva somente para os parâmetros da função *kernel*, pois o valor de C foi fixado em 10^4. A título de comparação, com a abordagem EME-SVM, foram também utilizados os outros modelos de ensemble apresentados nas seções 5.4 e 5.5. A Tabela 5.11 apresenta a média de desempenho, o erro quadrático médio de cada abordagem considerando 10 execuções.

Abordagem	Critério de Ordenação	Critério de Combinação	EQM	
			Nox	Price
SVM - ERBF	----	----	0,041822	0,034981
LS-SVM – RBF	-----	-----	0,042425	0,034984
Método Construtivo	EQM	Média Ponderada	0,039831	0,027542
Método Poda	EQM	Média Simples	0,039895	0,027739
GASel β = 0,4	----	Média Ponderada	0,037104	0,025213
EME-SVMc	EQM	SVM – RBF	0,026304	0,018991
EME-SVMp	EQM	SVM – RBF	0,027302	0,017890

Tabela 5.11 – Resultado para o conjunto de dados *Housing Boston*

Os resultados obtidos indicam que o EME-SVM pode promover uma seleção apropriada das SVMs e da função de *kernel*, isso para cada problema considerado,

representando assim uma especificação automática dos aspectos estruturais e paramétricos de projeto.

Além disso, a abordagem EME-SVM requer menos recursos computacionais quando comparada com qualquer variação de SVM, devido à necessidade de busca exaustiva associada a cada variação de SVM. Observe que a busca exaustiva para otimização de parâmetros de uma configuração de SVM isolada é responsável pela obtenção de um nível de desempenho similar àquele produzido pela abordagem EME-SVM, mas a um custo computacional maior.

5.6.4.1 Extensão de ensembles de SVMs para classificação com múltiplas classes

No Capítulo 4, foram apresentados vários métodos para aplicação de SVM a problemas com múltiplas classes, como, por exemplo, o método Um contra Todos e o método Um contra Um. Tais métodos podem ser igualmente aplicados a ensembles de SVMs. Nesta seção, os ensembles de SVMs serão estendidos para problemas de classificação com múltiplas classes. Dois tipos de extensão, de acordo com o nível de inserção dos ensembles de SVMs, serão propostos: (*i*) ensemble de SVMs no nível de classificação binária; e (*ii*) ensemble de SVM no nível de classificação com múltiplas classes.

Ensembles de SVMs no nível de classificação binária (conforme Figura 5.13) consistem de K ensembles de SVMs, no caso do método Um contra Todos, ou $K(K-1)/2$ ensembles de SVMs, no caso do método Um contra Um, onde K representa o número de classes. E cada ensemble de SVMs consiste de múltiplos SVMs treinados independentemente. Assim, um ensemble de SVMs é construído no nível de classificação binária (HE-SVM-B, do inglês *heterogeneous ensemble of support vector machines for binary classification*), ou seja, vários modelos de SVMs binários com *kernels* diferentes são treinados, selecionados por uma das técnicas apresentadas nas seções 5.4 e 5.5, e posteriormente agrupados, por exemplo, pelo voto majoritário. A decisão final é obtida a

partir das saídas dos vários ensembles de SVMs via estratégia de votação (para o método Um contra Um) ou através do valor máximo (para o método Um contra Todos).

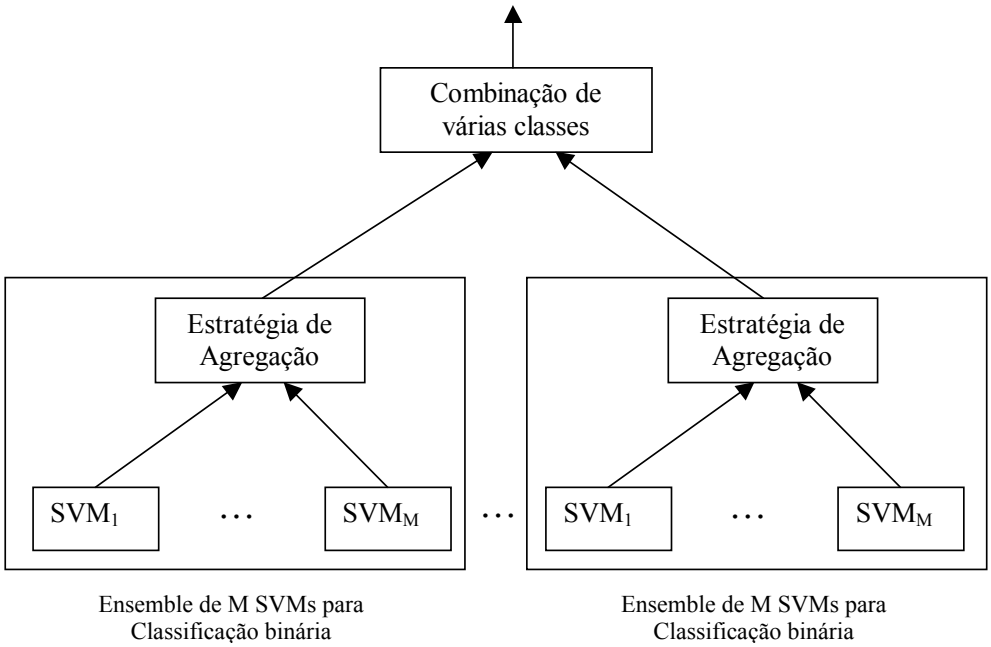


Figura 5.13 – Um ensemble de SVMs no nível de classificação binária

O ensemble de SVMs no nível de classificação com múltiplas classes consiste de *M* classificadores para múltiplas classes. E cada classificador consiste de *K* SVMs binários, no caso de método Um contra Todos, ou $K(K-1)/2$ classificadores binários, no caso do método Um contra Um. Assim, o ensemble de SVMs é construído no nível de classificação para múltiplas classes (HE-SVM-M, do inglês *heterogeneous ensemble of support vector machines for multiclass classification*). Cada classificador para múltiplas classes é constituído de vários SVMs com o mesmo *kernel*. A decisão final é obtida a partir da saída de vários classificadores para múltiplas classes via método de votação.

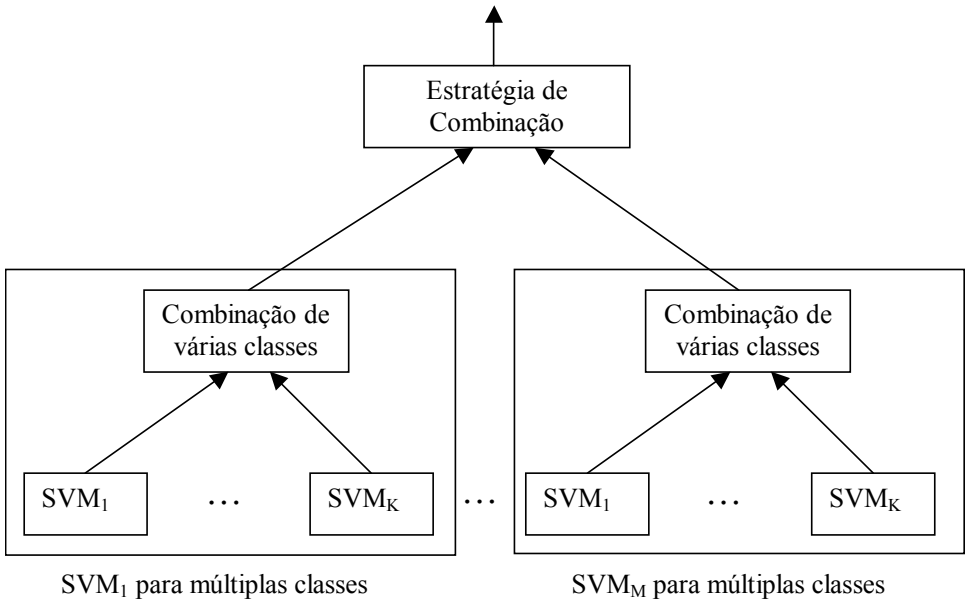


Figura 5.14 – Ensemble de SVMs no nível de classificação com múltiplas classes

Para avaliar a eficácia do ensemble de SVMs para problemas de classificação com múltiplas classes, dois problemas de classificação diferentes foram considerados: IRIS e Glass, ambos obtidos da base de dados UCI de aprendizado de máquina (BLAKE & MERZ, 1998).

O conjunto de dados da IRIS (FISHER, 1936) é um dos conjuntos de dados mais conhecidos encontrado na literatura de reconhecimento de padrões, contendo 3 classes, onde cada classe consiste de 50 amostras. Este conjunto já foi alvo de experimentos computacionais no Capítulo 4.

Por sua vez, o conjunto Glass, identificação do tipo de vidro, foi motivado por investigação criminal. Na cena do crime, o vidro deixado pode ser usado como evidência, se for corretamente classificado. O conjunto contém 6 classes, onde as classes contêm as seguintes quantidade de amostras: 70, 76, 17, 13, 9, 29, totalizando 214 amostras.

A título de comparação, foi utilizado o método Um contra Todos, Todos contra Todos e uma abordagem de treinamento de SVM para múltiplas classes GUERMEUR (2000), conforme descrito no Capítulo 4, seção 4.10.5. O parâmetro C foi fixado em 10^4 e o processo de validação cruzada com 10 partições foi utilizado para encontrar os parâmetros

dos *kernels*. O desempenho médio e o desvio padrão para tais abordagens são apresentados nas Tabelas 5.12 e 5.13.

Para o EME-SVM-B, o valor de *C* também foi fixado em 10^4, e os parâmetros dos *kernels* de seus componentes foram fixados nos valores padrões encontrados na literatura (Capítulo 4). O processo de seleção de componentes adotado foi o método construtivo, conforme descrito nas seções anteriores, e a estratégia de agregação das saídas dos componentes de cada ensemble foi realizada via voto majoritário. Devido à pequena quantidade de amostras, não serão apresentados resultados para a abordagem EME-SVM-M. As Tabelas 5.12 e 5.13, apresentam a média e o desvio padrão para 10 partições realizadas pelo processo de validação cruzada.

Tipo de *Kernel*	Um contra Todos		Um contra Um		Múltiplas classes	
	Parâmetro *kernel*	Taxa de Classificação incorreta	Parâmetro *kernel*	Taxa de Classificação incorreta	Parâmetro *kernel*	Taxa de Classificação incorreta
Linear	---	0,05333±0,023934	-----	0,02667±0,014741	----	0,1±0,030225
Polinomial	1	0,05333±0,023934	1	0,02667±0,014741	2,0	0,06±0,023201
RBF	16	0,04±0,022662	8	0,04±0,017778	0,5	0,04±0,017778
ERBF	64	0,046667±0,02	64	0,04±0,017778	1	0,04±0,017778
Tangente	1,1892	0,18667±0,038233	0,25	0,23333±0,037515	0,849	0,17333±0,022662
Fourier	1	0,82667±0,056394	0,25	1±0	0,25	0,68667±0,034498
Spline	----	0,05333±0,023934	----	0,05333±0,019373	----	0,06667±0,024343
Bn-spline	1	0,05333±0,019373	1	0,04±0,017778	1	0,04±0,017778
EME-SVM-B	-----	0,02752±0,025485	----	0,02667±0,014741	-----	----------

Tabela 5.12 – Resultados de classificação para o problema IRIS

Tipo de *Kernel*	Um contra Todos		Um contra Um		Múltiplas classes	
	Parâmetro *kernel*	Taxa de Classificação incorreta	Parâmetro *kernel*	Taxa de Classificação incorreta	Parâmetro *kernel*	Taxa de Classificação incorreta
Linear	-----	0,37208±0,047449	-----	0,34913±0,039606	----	0,41537±0,037481
Polinomial	2	0,36299±0,041910	2	0,29307±0,034128	1	0,41537±0,037481
RBF	2	0,30736±0,032573	2	0,29307±0,035693	2	0,38247±0,038192
ERBF	32	0,29351±0,038923	1	0,29351±0,035965	1	0,28874±0,033051
Tangente	2	0,59307±0,036246	0,25	0,30281±0,025071	0,25	0,63983±0,034821
Fourier	2,3784	0,66342±0,034927	0,25	0,69156±0,036979	3,3636	0,64913±0,037599
Spline	-----	0,29784±0,045784	-----	0,64437±0,043763	-----	0,39177±0,026845
Bn-spline		0,28774±0,035185		0,285120±0,03058		0,26082±0,028039
EME-SVM-B	-------	0,25682±0,028192	------	0,25628±0,029072	------	----------

Tabela 5.13 – Resultados de classificação para o problema Glass

A abordagem EME-SVM-B apresentou resultados predominantemente superiores às demais abordagens, indicando robustez e também economia de recursos computacionais na geração da solução, pela ausência de necessidade de busca exaustiva para sintonia de parâmetros, como se faz necessário nos demais casos.

5.7 Considerações Finais

Neste capítulo, foram apresentadas várias extensões e novas configurações de ensemble, tendo máquinas de vetores-suporte como componentes aplicados à solução de problemas de classificação e regressão. O objetivo desta extensão foi amenizar significativamente os efeitos negativos de especificações equivocadas de projeto. Os principais tópicos abordados foram os seguintes: (i) proposição de ensembles de SVMs heterogêneas, onde foi realizado um estudo da sensibilidade da abordagem proposta comparada com a melhor SVM em relação a mudanças na densidade amostral e aspectos paramétricos de projeto; (ii) proposição de novos critérios de ordenação para seleção de componentes baseados em dimensão VC e margem de separação; os quais foram comparados com critério de ordenação baseado no EQM, proposto por PERRONE & COOPER (1993); (iii) proposição de critérios de seleção de componentes baseados em algoritmos genéticos, cujo objetivo era diminuir a sensibilidade da abordagem proposta aos critérios de ordenação; (iv) extensão do conceito de ensemble de múltiplos estágios para SVM, sendo que aqui o objetivo foi propor um critério de combinação mais robusto baseado em trabalhos publicados na literatura e (*v*) extensão do conceito de ensemble para problemas de classificação com múltiplas classes.

Capítulo 6

Mistura de especialistas baseada em Máquinas de Vetores-Suporte

Resumo: A abordagem de mistura de especialistas (MEs) é composta de várias estruturas modulares, chamadas de especialistas, cujas respostas individuais são combinadas automaticamente por meio de uma outra estrutura, denominada *gating*. MEs são modelos de misturas que usam explicitamente uma estratégia dividir-e-conquistar, ou seja, aprendem a dividir automaticamente o espaço de entrada, sendo que sobreposições são possíveis entre as regiões resultantes e a cada região é atribuído um especialista, o qual pode ser implementado a priori, ou então juntamente com a *gating*, de forma iterativa. Pelas vantagens já apontadas em outros capítulos da tese junto a tarefas de classificação e regressão, máquinas de vetores-suporte podem ser tomadas como especialistas e como *gating* em uma arquitetura de mistura de especialistas. Neste capítulo, será apresentado um formalismo capaz de incorporar o problema de treinamento de máquinas de vetores-suporte como uma etapa do processo de síntese de misturas de especialistas. A formulação matemática resultante para o treinamento de um mistura de SVMs conduz a um problema quadrático que é similar àquele empregado no treinamento de uma única SVM, com nenhum aumento na dimensionalidade do problema quadrático associado. Em virtude da

existência de funções *kernel* com melhor desempenho nesta ou naquela região do espaço de entrada, a abordagem de mistura se mostra promissora no sentido de indicar apropriadamente junto a qual região do espaço de entrada cada SVM deve se concentrar e qual deve ser a parametrização associada a cada especialista. Experimentos computacionais apontam para a validade da proposta e permitem posicionar misturas de SVMs dentre as mais eficazes ferramentas de aprendizado de máquina de propósito geral.

6.1 Introdução

Sistema modulares permitem que o aprendizado de problemas complexos possa ser solucionado dividindo o problema em um subconjunto de problemas, cada um dos quais admitindo possivelmente uma maior tratabilidade que o problema original. Dentro do contexto de aprendizado supervisionado, arquiteturas modulares surgem quando os dados podem ser modelados localmente, dividindo assim o espaço de entrada em regiões. No Capítulo 3, foi descrito um tipo particular de sistema modular chamado de mistura de especialistas (ME).

O modelo de ME tradicional, introduzido por JORDAN & JACOBS (1994), tem um perceptron de uma camada com função de ativação *softmax* como *gating* e os especialistas com função de ativação linear (veja o Capítulo 3, para mais detalhes). Além desta, há duas variações chamadas mistura de especialistas localizados (LME) e mistura de especialistas *gated* (GME).

A primeira variante foi formulada por XU *et al*. (1995) e mais tarde desenvolvida por RAMAMURTI & GHOSH (1999). Ela realiza o problema da decomposição empregando funções de base radial hiper-elipsoidais produzidas via *kernels* normalizados. A segunda variante, formulada por WEIGEND *et al*. (1995), usa uma *gating* não-linear para coordenar a combinação dos especialistas não-lineares (tipicamente, perceptrons multicamadas).

A síntese deste tipo de modelo de mistura e suas variantes empregam explicitamente uma estratégia dividir-e-conquistar. A função-objetivo a ser otimizada é baseada na interpretação de MEs como modelos de densidade condicional (MCLACHLAN & BASFORD, 1988; JORDAN & JACOBS, 1994). Esta abordagem tem se mostrado particularmente útil na

atribuição dos especialistas para regimes diferentes em séries temporais estacionárias por partes (WEIGEND *et al.*, 1995) e modelagem de descontinuidades em problema de regressão e classificação (FRITSCH, 1996; MOERLAND, 1997; WATERHOUSE & ROBINSON, 1994).

No entanto, além de promover transições chaveadas entre regimes, os modelos de aprendizado de máquina resultantes podem implementar transições suaves e explorar automaticamente a diversidade de desempenho de regressores ou classificadores. Esta diversidade de desempenho se manifesta, por exemplo, quando são consideradas diferentes regiões do espaço de entrada e quando diferentes parâmetros são adotados para as funções *kernel*. Portanto, a incorporação de máquinas de vetores-suporte como especialistas, ou como a estrutura *gating* de uma mistura, se mostra promissora.

Além disso, como mencionado por JORDAN & JACOBS (1994), algoritmos dividir-e-conquistar geralmente tendem a apresentar um aumento na variância do erro, quando comparados com a abordagens não-modulares. Este fator pode conduzir a desempenho de generalização ruim, especialmente quando as entradas têm alta dimensionalidade e/ou a densidade amostral é baixa. Para abrandar este problema, JORDAN & JACOBS (1994) utilizaram funções lineares na *gating* e nos especialistas. Os algoritmos de treinamento usados, tipicamente, algoritmos baseados no gradiente ascendente (JACOBS *et al.*, 1991) ou algoritmos de maximização da esperança (EM) (JORDAN & JACOBS, 1994) são baseados na estimação do máximo da verossimilhança que é uma forma de minimização do risco empírico (VAPNIK, 1995).

As máquinas de vetores suporte (SVM), por sua vez, representam um procedimento de aprendizado não-paramétrico baseado na teoria de aprendizado estatístico (VAPNIK, 1995), sendo capazes de manipular de forma eficiente a flexibilidade dos modelos de classificação e regressão em espaços de elevada dimensão e na presença de densidade amostral baixa. A abordagem SVM é uma implementação do princípio de indução denominada minimização do risco estrutural. SVMs implementam integralmente o conceito de funções *kernel*, ou seja, mapeiam o vetor de entrada em um espaço de características de alta dimensionalidade e constroem hiperplanos ótimos de separação.

Diferentes funções *kernel* produzem diferentes SVMs. Logo, existe a possibilidade de haver regiões do espaço de entrada onde alguns *kernels* apresentam melhor capacidade de

generalização em relação a outros *kernels*. Isto justifica uma investigação de ferramentas capazes de alocar automaticamente os *kernels* a regiões mais promissoras do espaço de entrada.

Com o intuito de abordar problemas de classificação e regressão via mistura de especialistas, o objetivo é explorar as vantagens advindas de ambas as abordagens numa única ferramenta de aprendizado de máquina. Neste intuito, é apresentada uma formulação original que permite a incorporação de máquinas de vetores-suporte como módulos de arquiteturas de mistura, produzindo uma formulação matemática unificada. O resultado é uma mistura de especialistas baseada em máquinas de vetores suporte, denotada ME[SVM]. O problema quadrático resultante é similar àquele associado a uma única máquina de vetores-suporte, mantendo a mesma dimensão do problema de programação quadrática (QP). A abordagem ME[SVM], além de permitir o uso de diferentes especialistas em regiões diferentes do espaço de entrada, também suporta combinação de diferentes *kernels*, tais como *kernels* polinomiais e funções de base radial.

Os resultados de simulação a serem apresentados indicam que esta nova abordagem conduz a ganhos de desempenho em termos de generalização, quando comparada com uma única máquina de vetores-suporte. Outros resultados experimentais podem ser encontrados em LIMA *et al.* (2002; 2004).

6.2 Trabalhos Relacionados

A idéia de combinar (alguns autores também usam o termo de mistura neste sentido) muitas instâncias de SVMs dentro de uma mesma estrutura não é nova. Por exemplo, KWOK (1998) apresentou uma formulação que combinava várias instâncias de SVMs e cujo treinamento de toda a estrutura era baseada em um algoritmo de aprendizado similar ao adotado para o caso de uma única SVM. Tal abordagem, conforme já mencionado no Capítulo 4, tem grandes problemas relacionados à escalabilidade, uma vez que a dimensão do problema QP passa a depender tanto do número de amostras quanto do número de instâncias de SVMs. As SVMs já foram batizadas de especialistas, nesta formulação, mas não foi mencionado com seria modelada a *gating* e qual o tipo de treinamento a ser

realizado. Em outro trabalho, RIDA *et al*. (1999) propuseram: (*i*) dividir o conjunto de treinamento através de um algoritmo não-supervisionado para agrupamento dos dados; (*ii*) treinar um especialista (SVM) sobre cada subconjunto correspondente a um agrupamento; (*iii*) finalmente, combinar as saídas de todos os especialistas. Um esquema diferente foi proposto em COLLOBERT (2002), no qual cada especialista (SVM) é treinado sobre um conjunto de amostras escolhidas randomicamente a partir do conjunto de treinamento e, posteriormente, combinado via *gating* (um perceptron multicamada). De acordo com o desempenho apresentado pelos especialistas, algumas amostras do subconjunto atribuído a cada especialista podem ser realocadas a outro subconjunto. Neste caso, o processo de treinamento é re-iniciado.

Entretanto, nenhuma das abordagens mencionadas acima está de acordo com a idéia fundamental de modelos de mistura, já que não fornecem uma interpretação probabilística, nem para a saída da *gating*, nem para as saídas dos especialistas. Além disso, o treinamento desacoplado da *gating* e dos especialistas é realizado de maneira arbitrária e não atende explicitamente ao princípio dividir-e-conquistar.

Por outro lado, a abordagem ME[SVM] a ser apresentada admite uma interpretação probabilística e implementa explicitamente o princípio dividir-e-conquistar.

6.3 Formulação de Mistura de Especialistas baseado em SVM

De forma a realizar um desacoplamento no treinamento dos especialistas e da *gating*, um algoritmo baseado no critério de Maximização da Esperança (EM, do inglês *expectation maximization*) será utilizado, conforme descrito no Capítulo 3. O desenvolvimento da ME[SVM] consiste em introduzir junto ao modelo tradicional de MEs um termo de regularização na função custo de cada especialista, visando controlar a complexidade e melhorar o desempenho de generalização.

Após algumas manipulações algébricas, será mostrado que as máquinas de vetores-suporte, mesmo estando agora vinculadas à estrutura de uma mistura de especialistas, poderão ser treinadas empregando a formulação já adotada para o caso de uma única SVM, ou seja, com base em um problema quadrático específico. Esta forma de treinamento é

viabilizada pela implementação do desacoplamento no treinamento da *gating* e dos especialistas.

A formulação matemática vai envolver tanto a versão tradicional de SVM como a sua variação, chamada de LS-SVM. Ambas foram devidamente apresentadas no Capítulo 4. Procedimentos análogos podem ser realizados para outras variantes de máquinas de vetores-suporte, embora não serão apresentados aqui. Devido às peculiaridades envolvidas nos problemas de regressão e classificação, será dedicada uma subseção específica para cada um destes problemas.

6.4 Problemas de Regressão

A função-objetivo de mistura de especialistas é baseada na mistura de densidade de probabilidade condicional, conforme definido no Capítulo 3. Aplicando o algoritmo de maximização da esperança (EM) para esta estrutura, ocorre um desacoplamento do problema original em um subconjunto de problemas de maximização individuais, os quais podem ser solucionados independentemente. Portanto, uma função a ser maximizada para o *j*-ésimo especialista é dada por:

$$\theta_j^{k+1} = \arg\max_{\theta_j} \sum_{t=1}^{N} h_j^{(t)} \log P_j(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_j), \tag{6.1}$$

onde θ_j^{k+1} representa os parâmetros do *j*-ésimo especialista para a iteração *k*+1, $h_j^{(t)}$ a probabilidade a posteriori associada ao *j*-ésimo especialista, correspondente ao padrão *t*, e $P_j^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_j)$ é a densidade de probabilidade. No caso do problema de regressão com *d* saídas, $P_j^{(t)}(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_j)$ é suposta ser a densidade gaussiana, descrita abaixo:

$$P(\mathbf{y}^{(t)} \mid \mathbf{x}^{(t)}, \theta_j) = \frac{1}{(2\pi)^{1/2} \prod_{i=1}^{d} \sigma_{ji}} \exp\left\{ -\frac{1}{2} \sum_{i=1}^{d} \frac{(y_i^{(t)} - \mu_{ji}^{(t)})^2}{\sigma_{ji}^2} \right\}, \tag{6.2}$$

onde $\mu_{ji}^{(t)}$ é a *i*-ésima saída do *j*-ésimo especialista, e σ_{ji} é a variância associada com a *i*-ésima saída do *j*-ésimo especialista.

Sem perda de generalidade, considere o problema de regressão com uma única saída (isto é, $d = 1$) e com matriz de covariância diagonal. Logo, substituindo a expressão (6.2) em (6.1), obtém-se um funcional a ser maximizado, como segue:

$$\theta_j^{k+1} = \arg\max_{\theta_j} \left[-\sum_{t=1}^{N} \frac{h_j^{(t)}}{2\sigma_j} (y^{(t)} - \mu_j^{(t)})^2 - \left(\frac{1}{2} \log 2\pi - \log \sigma_j \right) \sum_{t=1}^{N} h_j^{(t)} \right]. \qquad (6.3)$$

Uma vez que é possível obter uma expressão para o cálculo de σ_j em somente um passo (veja o Capítulo 3 para maiores detalhes), a função-objetivo em (6.3) pode ser re-escrita para enfatizar somente sua dependência em relação aos parâmetros do especialista, produzindo:

$$\theta_j^{k+1} = \arg\max_{\boldsymbol{\theta}_j} \left[-\sum_{t=1}^{N} \frac{h_j^{(t)}}{2\sigma_j} (y^{(t)} - \mu_j^{(t)})^2 \right]. \qquad (6.4)$$

Na próxima seção, será apresentada uma formalização para obtenção dos parâmetros dos especialistas baseada na abordagem tradicional para SVM e na sua variante denominada LS-SVM.

6.4.1 Caso 1: SVM tradicional

Considere que a função μ_j de cada especialista pode ser representada por:

$$\mu_j = \sum_{i=1}^{p} c_i \varphi_i(\mathbf{x}) + b\,, \qquad (6.5)$$

onde $\{\varphi_i(\mathbf{x})\}_{i=1}^{p}$ é um dado conjunto de funções básicas linearmente independentes, e c_i e b são parâmetros a serem estimados a partir dos dados.

O problema de descoberta dos coeficientes c_i e b a partir do conjunto de dados é claramente mal condicionado, uma vez que o conjunto de possíveis soluções é infinito. A fim de tornar este problema bem condicionado, será adotada uma abordagem baseada na teoria de regularização (TIKHONOV & ARSENIM, 1997), a qual impõe uma restrição de suavidade à solução do problema de regressão. Portanto, ao invés de buscar uma solução

para a função-objetivo definida em (6.4), buscar-se-á uma solução para o seguinte problema de otimização:

$$\min_{\mu_j \in H} E[\mu_j] = C\sum_{t=1}^{N}\left(\frac{h_j^{(t)}}{2\sigma_j^2}\left(y^{(t)} - \mu_j^{(t)}\right)^2\right) + \frac{1}{2}\psi[\mu_j], \tag{6.6}$$

onde C é um número positivo, $\psi[\cdot]$ é um funcional de suavização e H é um conjunto de funções sobre os quais o funcional de suavização $\psi[\cdot]$ é definido. O primeiro termo do funcional em (6.6) força a saída da j-ésima amostra a ser próxima da saída desejada (princípio de minimização do risco empírico), e o segundo termo corresponde à suavidade, onde C é uma medida do compromisso entre estes dois termos. A expressão em (6.6) pode ser substituída por uma equivalente, na qual um conjunto de variáveis adicionais, chamadas variáveis de folga $\xi^{(t)}, \xi_*^{(t)}$, é introduzido produzindo:

$$\min_{\mu_j \in \mathrm{H}} E[\mu_j] = \frac{C}{2}\sum_{t=1}^{N}\pi_j^{(t)}\left((\xi^{(t)})^2 + (\xi_*^{(t)})^2\right) + \frac{1}{2}\psi[\mu_j], \tag{6.7}$$

onde $\pi_j^{(t)} = \dfrac{h_j^{(t)}}{\sigma_j^2}$ e sujeito a:

$$\begin{cases} \mu_j^{(t)} - y^{(t)} \le \varepsilon + \xi^{(t)}, & t = 1, \cdots, N \\ \xi^{(t)} \ge 0, & t = 1, \cdots, N \\ y^{(t)} - \mu_j^{(t)} - \le \varepsilon + \xi_*^{(t)}, & t = 1, \cdots, N \\ \xi_*^{(t)} \ge 0, & t = 1, \cdots, N \end{cases} \tag{6.8}$$

A equivalência entre os problemas (6.6) e (6.7) é estabelecida observando que no problema (6.7) uma penalidade é paga somente quando o valor absoluto do erro de interpolação excede ε. A fim de solucionar o problema de minimização com restrições, definido em (6.7), uma técnica baseada em multiplicadores de Lagrange será utilizada (veja Capítulo 4). O lagrangeano correspondente ao problema (6.7) é dado por:

$$\begin{aligned} L(\mu_j, \xi, \xi_*, \alpha, \alpha_*, \beta, \beta_*) = {} & \frac{C}{2}\sum_{t=1}^{N}\pi_j^{(t)}(\xi^{(t)})^2 + \frac{C}{2}\sum_{t=1}^{N}\pi_j^{(t)}(\xi_*^{(t)})^2 + \frac{1}{2}\psi[\mu_j] - \sum_{t=1}^{N}\alpha^{(t)}\left(y^{(t)} - \mu_j^{(t)} + \varepsilon + \xi^{(t)}\right) \\ & - \sum_{t=1}^{N}\alpha_*^{(t)}\left(-y^{(t)} + \mu_j^{(t)} + \varepsilon + \xi_*^{(t)}\right) - \sum_{t=1}^{N}\left(\beta^{(t)}\xi^{(t)} + \beta_*^{(t)}\xi_*^{(t)}\right) \end{aligned} \tag{6.9}$$

onde α, α_*, β, β_* são os multiplicadores de Lagrange. A solução do problema restrito acima pode ser obtida minimizando o lagrangeano (6.9) com respeito a μ_j (isto é, com relação a c_i e a b), ξ e ξ_*, e maximizando com respeito a α, α_*, β e β_*. Visto que a minimização é irrestrita, as derivadas com relação a c_i ,b ,ξ e ξ_* podem ser fixadas em zero, dando origem ao seguinte conjunto de equações:

$$\frac{\partial L}{\partial c_i} = 0 \rightarrow c_i = \sum_{t=1}^{N} (\alpha_*^{(t)} - \alpha^{(t)}) \varphi(\mathbf{x}^{(t)}) \tag{6.10}$$

$$\frac{\partial L}{\partial \xi^{(t)}} = 0 \rightarrow \beta^{(t)} + \alpha^{(t)} \leq \pi_j^{(t)} C \xi^{(t)} \quad t = 1, \cdots, N \tag{6.11}$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{t=1}^{N} \alpha_*^{(t)} = \sum_{t=1}^{N} \alpha^{(t)} \tag{6.12}$$

$$\frac{\partial L}{\partial \xi_*^{(t)}} = 0 \rightarrow \beta_*^{(t)} + \alpha_*^{(t)} \leq \pi_j^{(t)} C \xi_*^{(t)} \quad t = 1, \cdots, N \tag{6.13}$$

Substituindo a expansão (6.10) para os coeficiente c_i, no modelo do j-ésimo especialista (6.5), obtém-se uma solução da seguinte forma:

$$\mu_j(\mathbf{x}) = \sum_{t=1}^{N} (\alpha_*^{(t)} - \alpha^{(t)}) K(\mathbf{x}, \mathbf{x}^{(t)}) + b\,, \tag{6.14}$$

onde $K(\mathbf{x}, \mathbf{x}^{(t)}) = \left(\varphi(\mathbf{x}) \cdot \varphi(\mathbf{x}^{(t)})\right)$, conforme definido no Capítulo 4.

Substituindo as equações (6.12)-(6.14) no lagrangeano (6.9), obtém-se uma expressão em função de α e α_* que pode ser maximizada, sujeito às restrições adicionais listadas. Logo, obtém-se o seguinte problema:

$$\max_{\alpha, \alpha_*} L(\boldsymbol{\alpha}, \boldsymbol{\alpha}_*) = -\frac{1}{2}\left(\sum_{i,j=1}^{N} (\alpha_*^{(i)} - \alpha^{(i)})(\alpha_*^{(j)} - \alpha^{(j)}) K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) + \frac{1}{C} \sum_{i=1}^{N} \frac{(\alpha_*^{(i)})^2}{\pi_i} + \frac{1}{C} \sum_{i=1}^{N} \frac{(\alpha^{(i)})^2}{\pi_i} \right) - \sum_{i=1}^{N} \varepsilon_i (\alpha^{(i)} + \alpha_*^{(i)}) - \sum_{i=1}^{N} y^{(i)} (\alpha_*^{(i)} + \alpha^{(i)}) \tag{6.15}$$

sujeito às restrições: $\sum_{i=1}^{N} \alpha_*^{(i)} = \sum_{i=1}^{N} \alpha^{(i)}$, $\alpha_*^{(i)} \geq 0$, $\alpha^{(i)} \geq 0$, $i = 1, \cdots, N$,

Trata-se de um problema quadrático equivalente àquele formulado no Capítulo 4, a fim de calcular a solução para uma única máquina de vetores-suporte. Com esta nova formulação,

o treinamento de cada especialista na mistura pode ser definido em termos de um problema quadrático.

6.4.2 Caso 2: LS-SVM

A expressão (6.6) pode ser substituída pelo problema equivalente a seguir, no qual um conjunto de variáveis adicionais, $e_j^{(t)}$, diferentes daquelas mencionadas na seção anterior, será introduzido:

$$\min_{\mu_j \in H} E[\mu_j] = \frac{C}{2}\sum_{t=1}^{N} \pi_j^{(t)} (e_j^{(t)})^2 + \frac{1}{2}\psi[\mu_j], \tag{6.16}$$

onde $\pi_j^{(t)} = \dfrac{h_j^{(t)}}{\sigma_j^2}$ e sujeito a $\mu_j^{(t)} - y^{(t)} = e_j^{(t)} \qquad t = 1, \cdots, N$.

A fim de solucionar o problema de minimização com restrições acima, a mesma técnica baseada em multiplicadores de Lagrange da seção anterior pode ser utilizada. O lagrangeano correspondente para o problema acima é definido da seguinte forma:

$$L(\mu_j, e_j, \alpha, \beta) = \frac{C}{2}\sum_{t=1}^{N} \pi_j^{(t)} (e_j^{(t)})^2 + \frac{1}{2}\psi[\mu_j] - \sum_{t=1}^{N} \alpha^{(t)} \left(y^{(t)} - \mu_j^{(t)} + e_j^{(t)} \right), \tag{6.17}$$

onde $\alpha^{(t)}$ são os multiplicadores de Lagrange. A solução do problema com restrições acima se dá minimizando o lagrangeano (6.17) com relação a $\mu_j^{(t)}$ (isto é, com relação a c_i e a b) e a e_j, e maximizando com relação a $\alpha^{(t)}$. Uma vez que o passo de minimização é irrestrito, as derivadas podem ser fixadas em zero com relação a c_i e b, dando origem ao seguinte conjunto de equações:

$$\frac{\partial L}{\partial c_i} = 0 \rightarrow c_i = \lambda_i \sum_{t=1}^{N} \alpha^{(t)} \varphi(\mathbf{x}^{(t)}) \tag{6.18}$$

$$\frac{\partial L}{\partial e_j^{(t)}} = 0 \rightarrow \alpha^{(t)} = \pi_j^{(t)} C e_j^{(t)} \quad t = 1, \cdots, N \tag{6.19}$$

$$\frac{\partial L}{\partial b} = 0 \rightarrow \sum_{t=1}^{N} \alpha^{(t)} = 0 \tag{6.20}$$

$$\frac{\partial L}{\partial \alpha^{(t)}} = 0 \rightarrow y^{(t)} - \mu_j^{(t)} + e_j^{(t)} = 0 \;\; t = 1, \cdots, N \,. \tag{6.21}$$

Substituindo a expansão (6.18) para os coeficientes no modelo (6.5), obtém-se uma solução da seguinte forma:

$$f(\mathbf{x}) = \sum_{t=1}^{N} \alpha^{(t)} K(\mathbf{x}, \mathbf{x}^{(t)}) + b \,. \tag{6.22}$$

Substituindo as equações (6.20)-(6.22) no lagrangeano, obtém-se uma expressão em função de $\alpha^{(t)}$ que pode ser minimizada, sujeito às restrições adicionais listadas acima. Fazendo algumas manipulações algébricas, a solução para o problema acima pode ser obtida resolvendo o seguinte conjunto de equações lineares:

$$\begin{bmatrix} 0 & \mathbf{1}_v^T \\ \mathbf{1}_v & \mathbf{\Omega} + \mathbf{V}_c \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{Y} \end{bmatrix} \tag{6.23}$$

onde $\mathbf{Y} = [y^{(1)}; \ldots; y^{(N)}]$, $\mathbf{1}_v = [1, \ldots, 1]$, $\boldsymbol{\alpha} = [\alpha^{(1)}; \ldots; \alpha^{(N)}]$, e $\Omega_{kl} = \left(\varphi(\mathbf{x}^{(k)}) \cdot \varphi(\mathbf{x}^{(l)})\right)$ para $k, l = 1, \ldots, N$, e $\mathbf{V}_c = diag\left\{\frac{1}{C v_1}, \ldots, \frac{1}{C v_N}\right\}$.

Este é um sistema de equações lineares similar àquele formulado no Capítulo 4 para LS-SVM. Com esta nova formulação, o treinamento de cada especialista na mistura pode ser definido em termos de um conjunto de equações lineares.

6.4.3 Algoritmo para ME[SVM]

O seguinte pseudo-código será utilizado para avaliar a ME[SVM] proposta:

1. Inicialização da *gating*
   1.1. Se a *gating* for modelada por uma MLP, inicialize seus pesos no intervalo [−0,2;0,2];
   1.2. Se a *gating* for modelada por *kernels* normalizados,
      1.2.1. Escolha aleatoriamente alguns pontos para serem os centros das gaussianas;
      1.2.2. Inicialize a variância igual à variância dos dados de treinamento
2. Inicialização dos especialistas
   2.1. Use *K*-means ou processo randômico para dividir os dados em *m* partes, onde *m* é o número de especialistas;
   2.2. Treine cada especialista *j* sobre uma das *m* partições (resolvendo um problema quadrático na forma (6.15) ou um conjunto de equações lineares na forma (6.23)), considerado $\pi_j^{(t)} = 1, \quad t = 1, \ldots, N$;
3. Calcule a função de verossimilhança, definida no Capítulo 3;
4. Enquanto a mudança na verossimilhança for menor que um limiar (fixado nesta tese em 0,001)
   4.1. Passo E
      4.1.1. Para cada par $(\mathbf{x}^{(t)}, y^{(t)})$, calcule a probabilidade a posteriori $h_i^{(t)}$ usando os valores atuais;
      4.1.2. Calcule $\pi_j^{(t)}$.
   4.2. Passo M
      4.2.1. Para cada especialista *j*, solucione o problema quadrático definido em (6.15) ou o sistema de equações lineares definido em (6.23);
      4.2.2. Para a *gating*, solucione o problema de otimização associado, definido no Capítulo 3.
5. Volte ao passo 4, usando os valores dos parâmetros atualizados.

6.5 Experimentos computacionais para regressão

A seguir, serão apresentados alguns resultados envolvendo a abordagem ME-SVMs para problemas de regressão, predição de séries temporais e identificação de sistemas dinâmicos não-lineares. Os exemplos visam demonstrar a viabilidade da ferramenta em relação à abordagem tradicional, isto é, utilização de apenas uma SVM ou LS-SVM. Para os problemas de identificação de sistemas dinâmicos, a comparação estenderá a outros modelos clássicos de especialistas, tais como: modelos lineares e redes neurais artificiais. A partir do experimento no. 2, foi utilizado um critério para parar o treinamento da MEs ou

dos modelos individuais, chamado de critério de parada antecipado, ou seja, a cada passo de maximização (passo M no algoritmo EM) toda a arquitetura é testada. Caso o erro sobre o conjunto de validação esteja diminuindo, a arquitetura MEs é armazenada. Depois de um certo número de iterações definidas a priori, a arquitetura que obteve o menor erro sobre o conjunto de validação é testada.

6.5.1 Experimento no. 1

Este experimento visa demonstrar a viabilidade da abordagem ME[SVM] na alocação dos vários especialistas na forma de SVMs em regiões diferentes do espaço de entrada. A título de ilustração da habilidade desta ferramenta, será considerada a função $g(\mathbf{x}) = |\mathbf{x}|$, a qual será amostrada no intervalo $[-1,1]$. Com base nestes dados amostrados, foram obtidos duas soluções: uma via um único SVM e outra via uma ME[SVM], ambos tendo um *kernel* linear. O valor de C foi fixado em 10^4, e $\varepsilon = 0{,}01$. O resultado usando uma única SVM é apresentado na Figura 6.1. Apesar da simplicidade do problema, a utilização de uma única SVM com *kernel* linear não produziu resultados satisfatórios até porque a solução desejada é linear por partes. A Figura 6.2 apresenta a aproximação resultante para a abordagem ME[SVM]. Ao contrário da abordagem tradicional, a abordagem baseada em MEs consegue obter uma aproximação razoável para a função $g(\mathbf{x})$. Isto pode ser justificado analisando as Figuras 6.3 e 6.4. A Figura 6.3 apresenta as saídas para a *gating,* cada uma correspondente a um especialista. Como pode ser observado, a *gating* decompôs o problema em duas regiões distintas, atribuindo valor 1 para um especialista em um região e zero para o outro, com uma transição abrupta. Na outra região, foi realizado o oposto. A Figura 6.4, apresenta a saída de cada especialista, veja que a saída não foi alterada, continua sendo uma reta (devido ao *kernel* linear), mas cada especialista foi alocado a regiões diferentes.

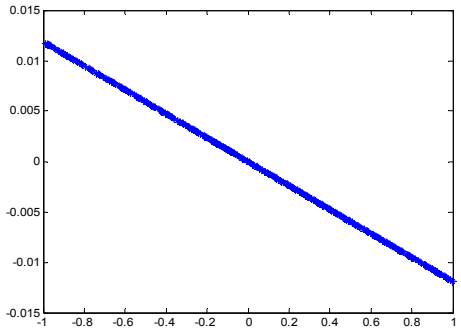


Figura 6.1 – Resultado da aproximação realizada pelo SVM com *kernel* linear.

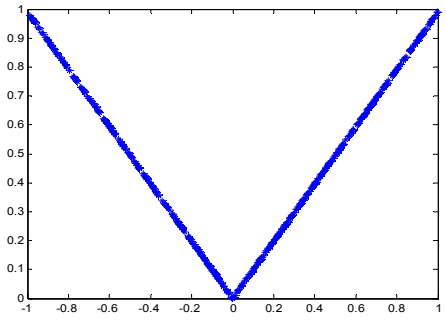


Figura 6.2 – Resultado da aproximação realizada pelo ME-SVMs com *kernel* linear.

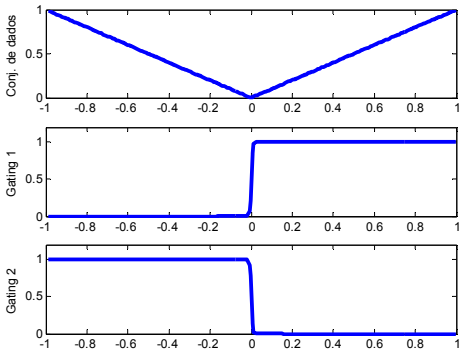


Figura 6.3 – Saída da *gating* da ME[SVM] para cada especialista. O primeiro gráfico contém a saída desejada.

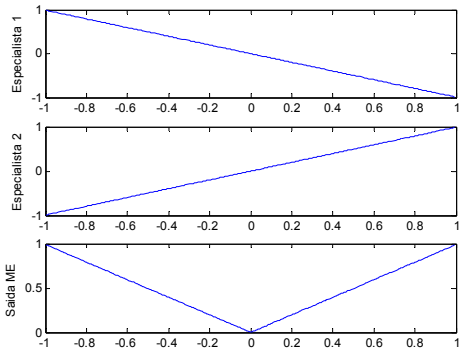


Figura 6.4 – Saída de cada especialista e a saída global da ME[SVM].

Uma questão decorrente dos resultados apresentados acima é a seguinte: o que acontecerá com a abordagem ME[SVM] se o *kernel* for não-linear, por exemplo, uma RBF. O resultado é apresentado nas Figuras 6.5 e 6.6. Ao contrário do *kernel* linear, o *kernel* RBF tem uma capacidade de modelagem maior. Na região onde ocorre à descontinuidade na derivada primeira, há uma não-linearidade que forçou a alocação de um outro especialista especificamente para esta região, o qual suavizou a transição realizada.

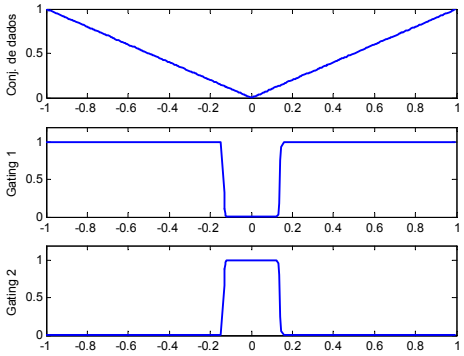


Figura 6.5 – Saída da *gating* da ME[SVM] para cada especialista. O primeiro gráfico contém a saída desejada.

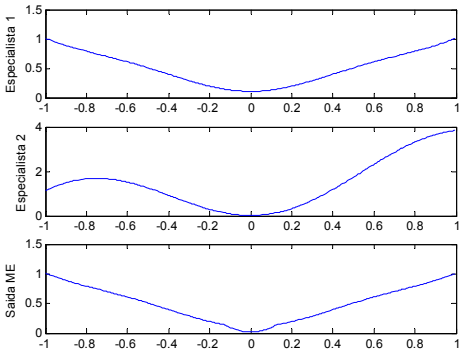


Figura 6.6 – Saída de cada especialista e a saída global da ME[SVM].

6.5.2 Experimento no. 2

Este experimento envolve a predição de séries temporais, com a série temporal sendo produzida artificialmente. O processo de geração dos dados está de acordo com a suposição da arquitetura de mistura de especialista. Neste caso, há dois processos:

$$x^{(t+1)} = 2 \cdot (1 - (x^{(t)})^2) - 1, \text{ se chaveamento} = 1 \qquad (6.24)$$

$$x^{(t+1)} = \tanh(-1{,}2 \cdot x^{(t)} + \xi^{(t+1)}); \; \xi \sim N(0;0{,}1), \text{ se chaveamento} = 0 \qquad (6.25)$$

O primeiro é um processo caótico. O segundo é um processo não-caótico ruidoso, o qual é uma composição de um processo auto-regressivo de ordem 1, com ruído gaussiano de variância igual a 0,1 (um nível de ruído relativamente alto, cujo desvio padrão é igual a 0,32). Um estudo detalhado de cada processo é apresentado em WEIGEND *et al*. (1995).

A dinâmica da série temporal é governada por um processo de Markov de primeira ordem. A matriz de transição de probabilidade de um processo para o outro é dada por um valor de 0,98 na diagonal (isto é, a probabilidade de permanecer no mesmo processo é fixado em 0,98) e 0,02 fora da diagonal (isto é, a probabilidade de chaveamento para outro processo é de 0,02). Tem-se então um tempo médio de chaveamento entre os processos de 50 passos. Somente o valor esperado é conhecido. O tempo exato quando ocorre o

chaveamento é randômico. Neste experimento, foram utilizados 1000 amostras para treinamento, 500 amostras para validação e 500 amostras para teste. A arquitetura consiste de dois especialistas e uma *gating*. O objetivo é predizer o próximo valor da série, $x^{(t+1)}$, ou seja, trata-se de um problema de predição de um único passo. O especialista tem acesso somente a dois valores atrasados da série $\{x^{(t-1)}, x^{(t)}\}$, enquanto a *gating* tem acesso aos quatro últimos valores da série como entrada $\{x^{(t-3)}, x^{(t-2)}, x^{(t-1)}, x^{(t)}\}$. O objetivo desta diferenciação é fornecer maior informação para a *gating* com o propósito de facilitar a alocação dos especialistas. Este número de entradas coincide com o utilizado por WEIGEND *et al*. (1995).

A *gating* será modelada por uma rede neural com 5 neurônios na camada escondida e função *softmax*. A Figura 6.7, apresenta a composição do processo e o correspondente chaveamento para o intervalo do conjunto de treinamento.

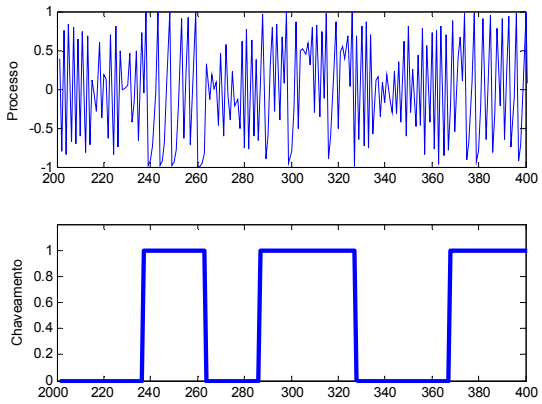


Figura 6. 7 – Na parte superior, é apresentada a composição dos dois processos. Na parte inferior, é apresentado o processo de chaveamento realizado para a obtenção da série temporal artificial.

O resultado obtido para a ME[SVM] com (*i*) dois especialistas SVM; (*ii*) *kernel* RBF; e (*iii*) parâmetro *C* fixado em 10^4, é apresentado na Figura 6.8. Na Figura 6.9, é apresentada a evolução do erro de treinamento, validação e teste. A partir de um certo valor (iteração igual a 24), o erro de validação começa a subir, indicando que a partir deste ponto

a arquitetura pode estar sofrendo sobre-ajuste referente aos dados de treinamento. Conseqüentemente, o treinamento deve ser finalizado.

A variância associada ao especialista que modelou o processo quadrático (Figura 6.10) é muito pequena (~0,001), limitada pelo valor a priori introduzido para evitar problemas numéricos. Já a variância associada com o especialista que modelou o processo ruidoso é próxima do nível de ruído daquele processo, isto é, 0,1 (veja Figura 6.10). O especialista 2 que se ocupou em modelar o processo ruidoso, estima sua variância um pouco abaixo do real, 0.06 (conforme Figura 6.10). Com isto, a arquitetura consegue fornecer uma estimativa da quantidade de ruído nos dados.

Compare o chaveamento real com aquele obtido pela saída da *gating* (veja Figura 6.11). Muito das vezes as saídas são binárias. Observe que a *gating* consegue uma boa aproximação para o chaveamento realizado pelo sistema de geração da série temporal.

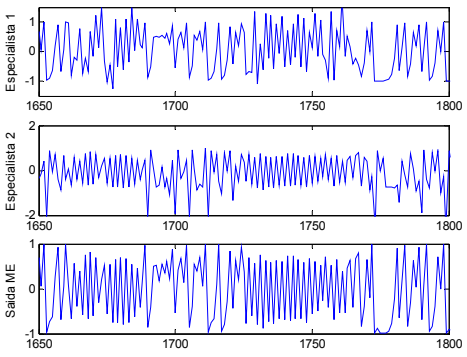


Figura 6.8 – Saída dos especialistas para o conjunto de teste.

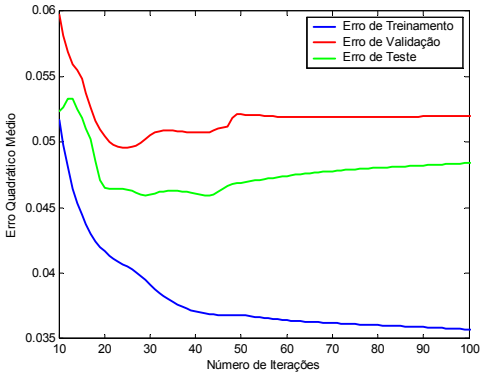


Figura 6.9 – Evolução dos erros de treinamento, validação e teste.

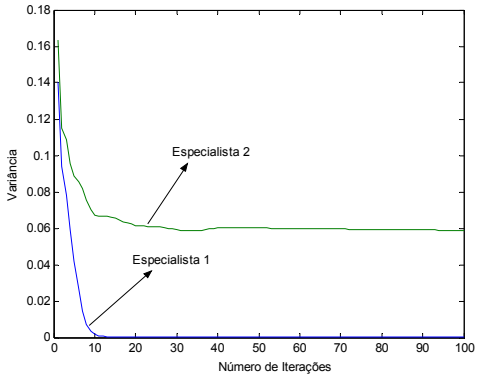


Figura 6.10 – Evolução da variância durante o treinamento.

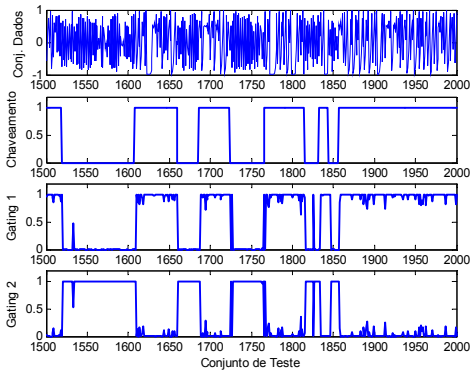


Figura 6.11 – Saída da *gating* para o conjunto de teste.

6.5.3 Experimento no. 3

Neste terceiro experimento, o objetivo é aplicar a arquitetura ME[SVM] à predição da série temporal laser do *Santa Fe Time Series Prediction and Analysis Competition* (WEIGEND & GERSHENFELD, 1994). A série laser é uma série estacionária e seu comportamento pode ser aproximado através de um conjunto de três equações a diferenças não-lineares, chamadas de equações de Lorenz.

O objetivo é predizer o próximo valor, $x^{(t+1)}$, ou seja, trata-se de um problema de predição de um único passo. As entradas para cada especialista são os 10 valores mais recentes da série temporal $\{x^{(t-9)}, x^{(t-8)}, \ldots, x^{(t)}\}$ (WEIGEND, 1995). Cada especialista é composto de uma SVM com *kernel* RBF e o valor de C é fixado em 10^4. A rede *gating* é uma MLP com 5 neurônios na camada escondida, com função *softmax* na saída, a qual recebe os mesmos 10 valores de entrada dos especialistas. Foram obtidas 2000 amostras, sendo 900 amostras correspondes ao conjunto de treinamento, 100 amostras para o conjunto de validação e as restantes 1000 amostras para o conjunto de teste.

Não é conhecido a priori o número ótimo de especialistas para este problema. Tendo este objetivo em mente, foram realizadas várias simulações com o número de especialistas variando entre 2, 3 e 5. Em grande parte das simulações, os resultados convergiram para 3

especialistas, isto é, a saída da *gating* para os outros especialistas era aproximadamente igual a zero, fazendo com que a contribuição destes para a predição fosse bastante pequena. A Figura 6.12 ilustra o comportamento da saída da *gating* para o conjunto de teste, para 2 especialistas. Observe que, na região compreendida entre 1100 e 1200, existe somente um único especialista contribuindo para a solução. No entanto, o mesmo não ocorreu para outras regiões. Já na Figura 6.13, é apresentada a saída para os 2 especialistas.

Nas figuras 6.14 e 6.15, são apresentados resultados de predição para uma única SVM com *kernel* RBF e para ME[SVM] com *kernel* RBF. Observe que a predição para ambas as abordagens foram excelentes. O erro quadrático médio alcançado no conjunto de teste foi de 0,000865 para a abordagem ME[SVM], enquanto que para SVM com *kernel* RBF obteve-se 0,001005, considerando a série normalizada no intervalo [−1;1]. Apesar do desempenho da ME[SVM] não ter sido tão destacado frente aquele obtido com modelos não-modulares, este experimento visa demonstra a viabilidade de tal abordagem para problemas de predição de séries temporais.

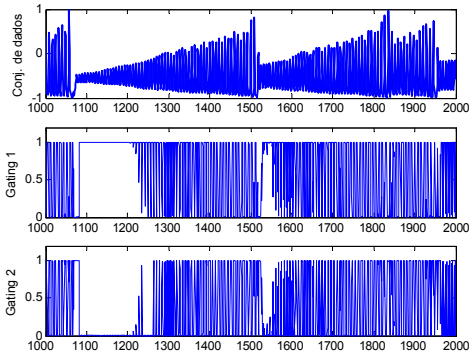


Figura 6.12 – Saída da *gating* sobre parte do conjunto de teste.

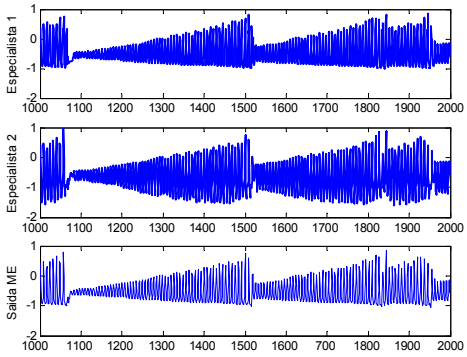


Figura 6.13 – Saída dos especialistas sobre parte do conjunto de teste e saída da mistura.

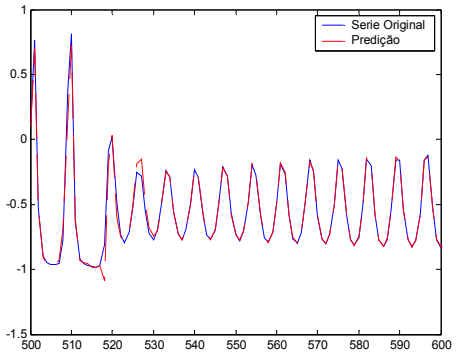


Figura 6.14 – Predição realizada por uma SVM com *kernel* RBF.

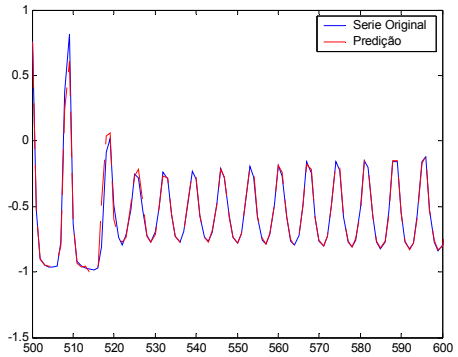


Figura 6.15 – Predição realizada por uma ME[SVM] com *kernel* RBF.

6.5.4 Identificação de sistemas não-lineares

Nesta seção, a abordagem ME[SVM] será aplicada a quatro problemas de identificação de sistema. Em cada análise, três conjuntos de dados foram gerados: um para treinamento; outro para validação do modelo; e outro para testar o desempenho da ME. Além disso, em todos os experimentos foram realizadas 10 execuções para cada estratégia. Os resultados apresentados correspondem à melhor configuração de ME (ou única configuração de SVM, rede neural) encontrada. Os dados de entrada foram sempre pré-processados para se ajustarem ao intervalo [–1;1]. Para cada MLP, 5000 iterações foram fixadas como critério de parada, 500 iterações foram usadas para a maximização do passo M (quando usando uma rede neural tanto como especialista quanto como *gating*) e 500 iterações para o passo EM em todos os casos.

Algumas considerações com relação à planta a ser identificada:

a. As plantas apresentam dinâmica discreta;
b. As plantas podem expressar comportamento linear e/ou não-linear;
c. Somente dados de entrada e saída são considerados;
d. A planta pode ser variante ou invariante no tempo.

A Figura 6.16 apresenta a arquitetura de identificação utilizada para modelar as plantas consideradas, onde o objetivo é minimizar a diferença entre a saída da planta $y(k+1)$ e a saída gerada pelo modelo $\hat{y}(k+1)$. LIMA *et al*. (2002) realizou um estudo detalhado de identificação de sistemas utilizando algumas variações de MEs.

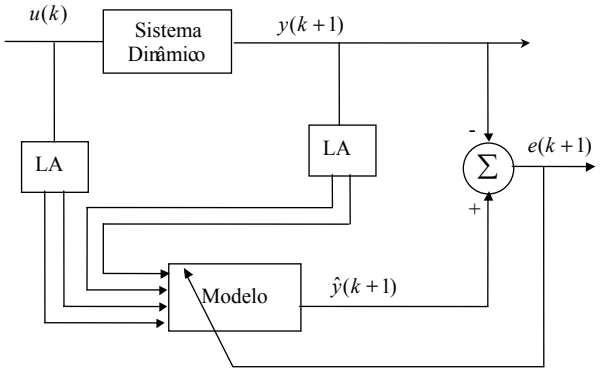


Figura 6.16 – Arquitetura de identificação de sistema não-linear (LA – linhas de atraso)

6.5.4.1 Problema no. 1

Neste experimento, a planta a ser modelada é descrita por:

$$y(k+1) = \frac{y(k)y(k-1)[y(k)-2{,}5]}{1+y^2(k)+y^2(k-1)} + u(k)$$

As entradas foram randomicamente distribuídas no intervalo [−2;2]. O objetivo é estimar o valor de $y(k+1)$ baseado nos valores de $y(k)$, $y(k-1)$ e u(k). Foram geradas 1000 amostras para treinamento e 500 amostras para validação do modelo. Após obter um modelo para o sistema, este é testado sobre duas entradas de controle diferentes, a saber: $u_1(k) = sen(2\pi k/25)$, com 120 amostras, e $u_1(k) = 1{,}6 \cdot sen(2\pi k/30)$, com 180 amostras. O resultado para a arquitetura ME[SVM] frente ao primeiro conjunto de teste é apresentado nas Figuras 6.17, 6.18, 6.19 e 6.20. A Tabela 6.0 apresenta os melhores resultados para algumas variações, tanto nos especialista quanto na *gating*. Os valores em negrito correspondem aos menores erros quadráticos médio obtidos. Observe que o melhor resultado foi obtido considerando ME[SVM] com *kernel* RBF para um conjunto de teste e *kernel* ERBF para o outro conjunto.

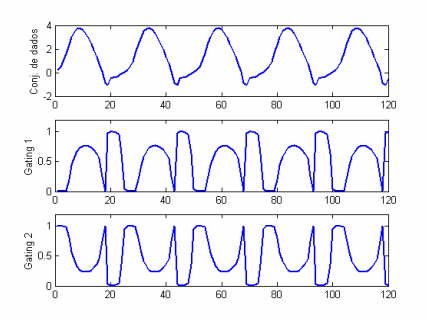


Figura 6.17 – Saída da rede *gating* de uma ME[SVM] com *kernel* RBF.

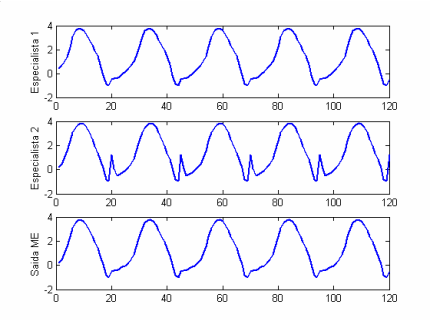


Figura 6.18 – Saída dos especialistas de ME[SVM] com *kernel* RBF e saída da mistura.

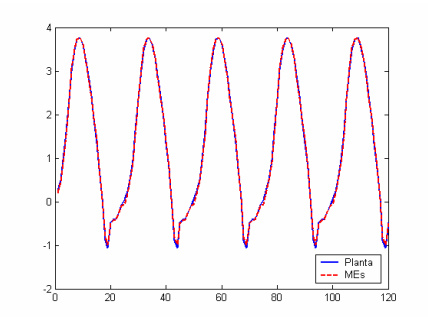


Figura 6.19 – Saída da planta para o conjunto de teste versus saída estimada.

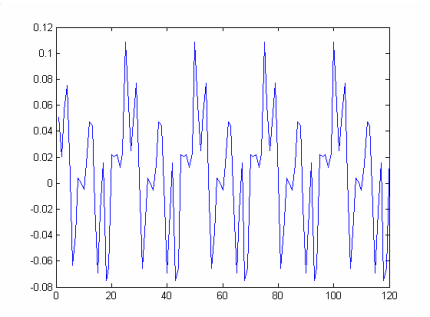


Figura 6.20 – Erro para arquitetura ME[SVM] com *kernel* RBF, considerando o primeiro conjunto de teste.

Abordagens		*m*	*nh*	Erro Quadrático Médio		
Gating	Especialista			Treinamento	Teste 1	Teste 2
Kernels Nomalizados	Linear	2	----	0,124747	0,123594	0,168098
		3	----	0,047920	0,030875	0,079442
		5	----	0,012723	0,008683	0,021301
		10	----	0,004402	0,001493	0,001744
	MLP	2	3	0,021091	0,027110	0,058830
		2	5	0,005158	0,004408	0,008884
	SVM – RBF	2	----	0,002516	0,001469	0,002256
	SVM - ERBF	2	----	0,002818	0,001358	0,002345
MLP	MLP	2	3	0,008503	0,013124	0,021882
		2	5	0,002237	0,001387	0,002618
	SVM – RBF	2	----	0,001685	0,002082	**0,001481**
	SVM - ERBF	2	----	0,001725	**0,001235**	0,001924
Única MLP		----	6	0,006593	0,006627	0,019326
		----	10	0,006176	0,004633	0,004716
		----	15	0,005676	0,007138	0,024625
Único SVM – RBF		----	----	0,004568	0,003567	0,003678
Único SVM – ERBF		----	----	0,004366	0,003895	0,004356

Tabela 6.1 – Resultados comparativos para todas as abordagens. Os valores em negrito indicam os melhores desempenhos, *m* indica o número de especialistas e *nh* o número de neurônios ocultos da MLP.

6.5.4.2 Problema no. 2

Neste experimento, a planta a ser modelada é descrita por:

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k).$$

As entradas foram randomicamente distribuídas no intervalo [–2;2]. O objetivo é estimar o valor de $y(k+1)$ baseado nos valores de $y(k)$ e $u(k)$. Foram geradas 2000 amostras para treinamento e 1000 amostras para validação do modelo, Após obter o modelo para o sistema, este é testado considerando a entrada de controle dada por: $u(k) = sen(2\pi k/25) + sen(2\pi k/30)$, com 100 amostras. Os resultado para a arquitetura ME[SVM] para o conjunto de teste são apresentados nas Figuras 6.20, 6.21, 6.22 e 6.23. A Tabela 6.1 apresenta os melhores resultados para todas as estratégias. Os valores em negrito correspondem aos menores erros quadráticos obtidos. Ao contrário do experimento anterior, a abordagem ME[SVM] com 2 especialistas não foi capaz de produzir resultados equivalentes àqueles produzidos pelas MEs com 5 especialistas como redes neurais.

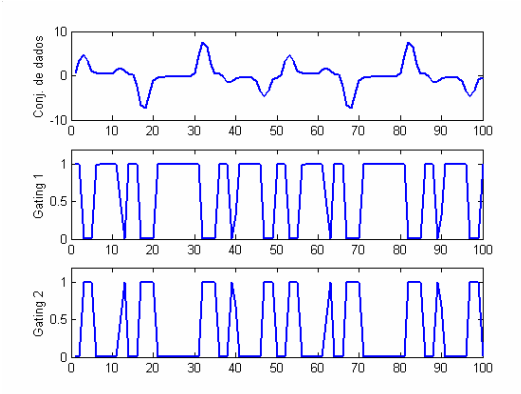


Figura 6.21 – Saída da rede *gating* de uma ME[SVM].

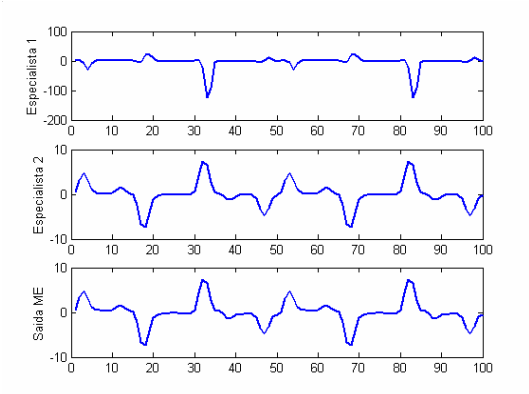


Figura 6.22 – Saída dos especialistas da ME[SVM] e saída da mistura.

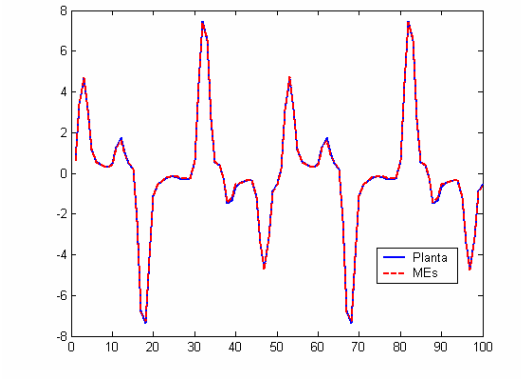


Figura 6.23 – Saída da planta para o conjunto de teste versus estimada.

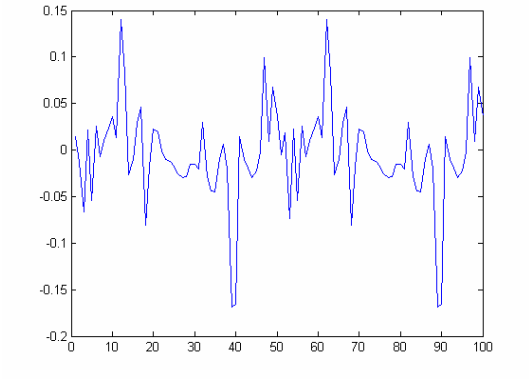


Figura 6.24 – Erro frente ao primeiro conjunto de teste para a arquitetura ME[SVM].

Abordagens		*m*	*nh*	Erro Quadrático Médio	
Gating	Especialista			Treinamento	Teste
Kernels Nomalizados	Linear	2		0,759738	0,494344
		3		0,153603	0,149417
		5		0,112203	0,104603
		10		0,051083	0,030911
	MLP	2	3	0,025556	0,023278
		2	5	0,003130	0,002473
	SVM – RBF	2		0,002155	0,001669
	SVM - ERBF	2		0,002360	0,001317
MLP	MLP	2	3	0,026901	0,024088
		2	5	**0,001284**	**0,001111**
	SVM – RBF	2		0,001845	0,001556
	SVM - ERBF	2		0,002045	0,001245
Única MLP		----	6	0,022750	0,027032
		----	10	0,024882	0,030676
		----	15	0,020760	0,026027
Único SVM – RBF		----	----	0,005578	0,009987
Único SVM – ERBF		----	----	0,006789	0,008754

Tabela 6.2 – Resultados comparativos para todas as abordagens. Os valores em negrito indicam os melhores desempenhos, *m* indica o número de especialistas e *nh* o número de neurônios ocultos da MLP.

6.5.4.3 Problema no. 3

Neste experimento, a planta a ser modelada é descrita por:

$$y(k+1) = \frac{y(k)y(k-1)y(k-2)u(k-1)(y(k-2)-1))+u(k)}{1+y^2(k-1)+y^2(k-2)}$$

As entradas foram randomicamente distribuídas no intervalo [−1;1]. O objetivo é estimar o valor de $y(k+1)$ baseado nos valores de $y(k)$, $y(k-1)$, $y(k-2)$, $u(k)$ e $u(k-1)$. Foram geradas 2000 amostras para treinamento e 1000 amostras para validação do modelo. Após obter o modelo para o sistema, este é testado considerado a seguinte entrada de controle:

$$u(k) = \begin{cases} sen(2\pi k/250) & se \quad 0 \le k \le 500 \\ 0{,}8 \cdot sen(2\pi k/250) + 0{,}2 \cdot \sin(2\pi k/25) & se \quad k \ge 500 \end{cases}$$

com 1000 amostras. Os resultados para a arquitetura ME[SVM], referentes ao conjunto de teste, são apresentados nas Figuras 6.24, 6.25, 6.26 e 6.27. A Tabela 6.2 apresenta os melhores resultados para todas as estratégias. Os valores em negrito correspondem aos menores erros quadráticos obtidos.

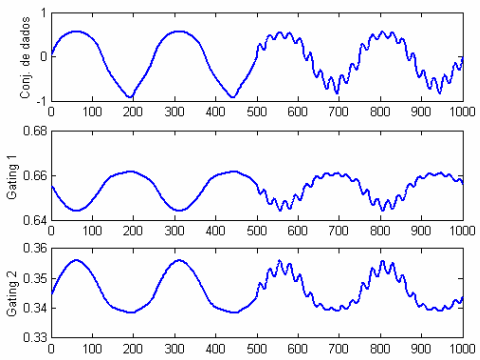


Figura 6.25 – Saída da rede *gating* da ME[SVM].

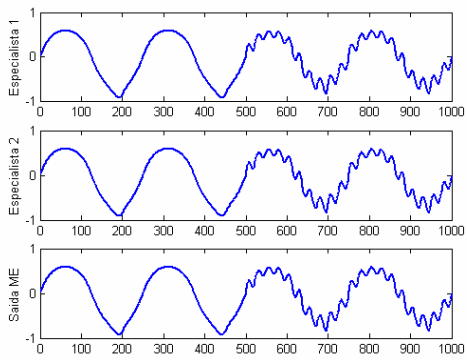


Figura 6.26 – Saída dos especialistas da ME[SVM] e saída da mistura.

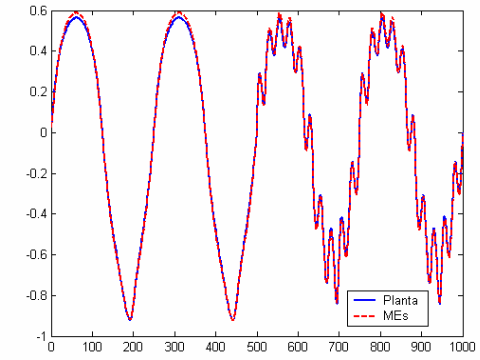


Figura 6.27 – Saída da planta para o conjunto de teste versus saída estimada.

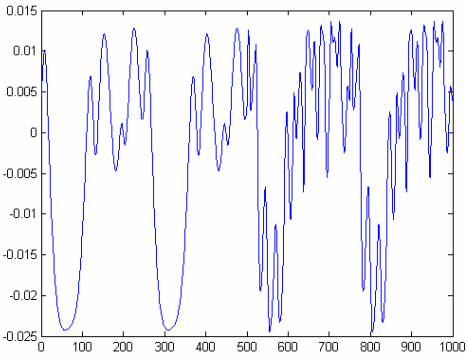


Figura 6.28 – Erro para arquitetura ME[SVM] com *kernel* RBF, considerando o conjunto de teste.

Abordagens		*m*	*nh*	Erro Quadrático Médio	
Gating	Especialista			Treinamento	Teste
Kernels Nomalizados	Linear	2	----	0,077161	0,006187
		3	----	0,066640	0,003067
		5	----	0,048397	0,000977
		10	----	0,029592	0,003301
	MLP	2	3	0,035879	0,001079
		2	5	0,019010	0,002042
	SVM – RBF	2	----	0,009546	0,000178
	SVM - ERBF	2	----	0,009678	0,000157
MLP	MLP	2	3	0,026984	0,000374
		2	5	0,014455	0,001720
	SVM – RBF	2	----	**0,008354**	**0,000146**
	SVM - ERBF	2	----	0,009346	0,000216
Única MLP		----	6	0,035331	0,008275
		----	10	0,029025	0,003588
		----	15	0,027594	0,002001
Único SVM – RBF		----	----	0,014789	0,001456
Único SVM – ERBF		----	----	0,015679	0,001346

Tabela 6.3 – Resultados comparativos para todas as abordagens. Os valores em negrito indicam os melhores desempenhos, *m* indica o número de especialistas e *nh* número de neurônios ocultos da MLP.

6.5.4.4 Problema no. 4

Neste experimento, a planta a ser modelada é descrita por:

$$y(k+1) = \frac{5 \cdot y(k)y(k-1)}{1+y(k)^2+y(k-1)^2+y(k-2)^2} + u(k) - 0{,}8 \cdot u(k-1)$$

As entradas foram randomicamente distribuídas no intervalo [–2;2]. O objetivo é estimar o valor de $y(k+1)$ baseado nos valores de $y(k)$, $y(k-1)$, $y(k-2)$, $u(k)$ e $u(k-1)$. Foram geradas 1000 amostras para treinamento e 500 amostras para validação do modelo. Após ter sido obtido o modelo para o sistema, este é testado sobre duas entradas de controle diferentes, a saber: $u_1(k) = sen(2\pi k/25)$, com 100 amostras, e $u_1(k) = sen(2k\pi/30) + sen(2k\pi/10)$, com 180 amostras. Os resultados para a arquitetura ME[SVM], referentes ao primeiro conjunto de teste, são apresentados nas Figuras 6.28, 6.29, 6.30 e 6.31. A Tabela 6.3 apresenta um resumo dos resultados envolvendo todas as estratégias.

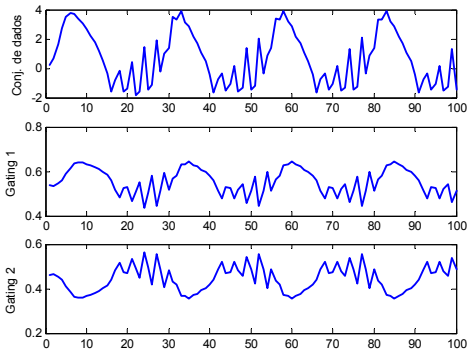


Figura 6.29 – Saída da rede *gating* da ME[SVM].

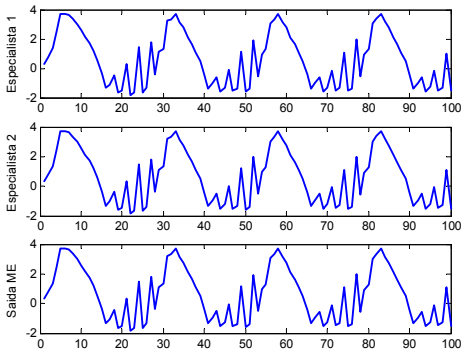


Figura 6.30 – Saída dos especialistas da ME[SVM] e saída da mistura.

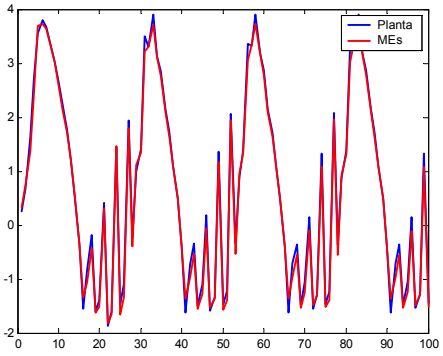


Figura 6.31 – Saída da planta para o conjunto de teste versus saída estimada.

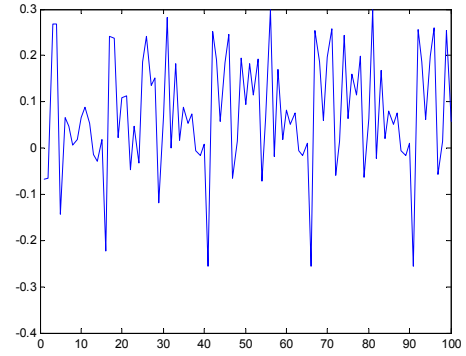


Figura 6.32 – Erro para arquitetura ME[SVM] com *kernel* RBF, considerando o primeiro conjunto de teste.

Abordagens		*m*	*nh*	Erro Quadrático Médio		
Gating	Especialista			Treinamento	Teste 1	Teste 2
Kernels Nomalizados	Linear	2	----	0,326431	0,332603	0,344497
		3	----	0,220246	0,247611	0,181566
		5	----	0,090607	0,086472	0,087474
		10	----	0,049561	0,062444	0,045242
	MLP	2	3	0,086098	0,118169	0,098534
		2	5	0,026953	0,013514	0,026860
	SVM – RBF	2	----	0,022352	**0,014564**	**0,021315**
	SVM - ERBF	2	----	0,021256	0,016548	0,022456
MLP	MLP	2	3	0,075632	0,094423	0,100140
		2	5	0,020341	0,017269	0,036195
	SVM – RBF	2	----	0,018425	0,013515	0,022415
	SVM - ERBF	2	----	**0,017845**	**0,013314**	**0,022312**
Única MLP		----	6	0,073053	0,094233	0,145163
		----	10	0,066721	0,089364	0,107910
		----	15	0,054744	0,072087	0,073008
Único SVM – RBF		----	----	0,020156	0,054565	0,071846
Único SVM – ERBF		----	----	0,018478	0,068364	0,085563

Tabela 6.4 – Resultados comparativos para todas as abordagens. Os valores em negrito indicam os melhores desempenhos, *m* indica o número de especialistas e *nh* o número de neurônios ocultos da MLP.

6.6 Problemas de classificação

Conforme visto no Capítulo 3, a função-objetivo para o especialista *j*, correspondente ao problema de maximização da verossimilhança, pode ser escrita como:

$$\theta_j = \arg\max \sum_{t=1}^{N} \sum_{l=1}^{K} h_{ij}^{(t)} y_l^{(t)} \log P(\mathbf{y}^{(t)} |, \mathbf{x}^{(t)}, \theta_j), \tag{6.26}$$

onde $P(\mathbf{y}^{(t)} |, \mathbf{x}^{(t)}, \theta_j)$ é a densidade condicional e representa o modelo probabilístico da mistura de especialistas. Na literatura, se destacam dois tipos de densidade condicional, a saber: densidade condicional multinomial e densidade de Bernoulli generalizada.

Usando o modelo de densidade multinomial (definido no Capítulo 3), obtém-se a seguinte função-objetivo para o especialista *j*, a ser maximizada na iteração *k*+1:

$$\theta_j^{(k+1)} = \arg\max \sum_{t=1}^{N} \sum_{l=1}^{K} h_j^{(t)} y_l^{(t)} \log \mu_{j_l}^{(t)}, \tag{6.27}$$

onde K é o número de classes e μ_{jl} é a l-ésima saída do especialista j, a qual pode ser representada na forma:

$$\mu_{jl} = \frac{\exp(s_{jl})}{\sum_{i=1}^{K} \exp(s_{ji})}, \tag{6.28}$$

e s_{jl} é a l-ésima saída do especialista j antes de aplicar a função *softmax* da equação (6.28). Aplicando logaritmo em ambos os lados da equação (6.28), obtém-se:

$$s_{jl}^{(t)} = \ln \mu_j^{(t)} + \ln \sum_{i=1}^{K} \exp(s_{ji}^{(t)}), \tag{6.29}$$

onde o segundo termo é constante para todo $s_{jl}^{(t)}$ (l = 1,...,K) e pode ser desprezado quando é aplicada a função *softmax* da equação (6.28) novamente. Portanto, das equações (6.27) e (6.28), obtém-se:

$$\theta_j^{(k+1)} = \arg\max \sum_{t=1}^{N} \sum_{l=1}^{k} h_j^{(t)} y_l^{(t)} s_{jl}^{(t)} . \tag{6.30}$$

Voltado para problemas de classificação, as próximas seções irão apresentar algumas possíveis soluções para a equação (6.30) com base em SVMs e/ou LS-SVMs. O objetivo é explorar em uma única ferramenta de aprendizado de máquina a habilidade de generalização da abordagem SVM e a capacidade da abordagem ME de dividir automaticamente problemas complexos em problemas relativamente simples. Os resultados de simulação indicam que a abordagem de mistura de especialistas é capaz de amenizar muitos dos problemas relacionados com a abordagem SVM, tais como a seleção dos parâmetros do *kernel* e a escolha de um valor para o parâmetro C.

6.6.1 Caso 1: SVM tradicional

A obtenção de uma solução para a equação (6.30) é equivalente a solucionar o seguinte problema, que envolve um conjunto adicional de variáveis, na forma:

$$\min_{\mathbf{w}_k, b_k, \xi_k} \; J(\mathbf{w}_k, b_k, \xi_k^{(t)}) = \sum_{t=1}^{N} h_j^{(t)} \sum_{i=1}^{K} \xi_i^{(t)}, \tag{6.31}$$

$$\text{sujeito a} \begin{cases} \bar{y}_1^{(t)}[w_1^T \varphi(\mathbf{x}^{(t)}) + b_1] \geq 1 - \xi_1^{(t)}, t = 1, \ldots, N \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \bar{y}_K^{(t)}[w_K^T \varphi(\mathbf{x}^{(t)}) + b_K] \geq 1 - \xi_K^{(t)}, t = 1, \ldots, N \end{cases}, \tag{6.32}$$

onde $\bar{y}_k^{(t)} \in \{\pm 1\}$ denota a *k*-ésima saída desejada para o padrão *t*. Definindo:

$$s_{jl}(\mathbf{x}) = \sum_{i=1}^{K} \beta_i [\mathbf{w}_l^T \varphi(\mathbf{x}) + b_l], \tag{6.33}$$

onde $\{\varphi(\mathbf{x})\}$ é um conjunto de funções-base linearmente independente, $\mathbf{w}_l$ e b_l (l = 1,...,K) são os parâmetros a serem estimados a partir do conjunto de dados de treinamento. Repare que os parâmetros β_i, i = 1,...,K, podem ser interpretados como responsáveis por uma mudança na codificação, uma vez que a SVM trabalha com codificação real e as MEs com codificação binária. Sendo assim, β_i, i = 1,...,K, transformam a codificação Um contra Todos (usada para problemas com múltiplas classes) em K codificações binárias. Os parâmetros β_i, i = 1,...,K, podem ser obtidos via quadrados mínimos (veja Figura 6.33).

Encontrar os coeficientes $\mathbf{w}_l$ e b_l a partir dos dados de treinamento é um problema considerado mal condicionado, uma vez que há um número infinito de soluções possíveis. Da mesma forma que no problema de regressão, pode-se impor uma restrição adicional de suavidade à função-objetivo do problema de classificação (6.31), conforme proposto pela teoria de regularização (TIKHONOV & ARSENIM, 1977). Adotando tal restrição, obtém-se o seguinte problema:

$$\min_{\mathbf{w}_k, b_k, \xi_k} J(\mathbf{w}_k, b_k, \xi_k^{(t)}) = \frac{1}{2} \sum_{i=1}^{K} \mathbf{w}_k^T \mathbf{w}_k + \sum_{t=1}^{N} h_j^{(t)} \sum_{i=1}^{K} \xi_i^{(t)}, \tag{6.34}$$

sujeito às restrições definidas em (6.32). Para solucionar o problema de otimização restrita acima, será utilizada a representação no espaço dual. Fazendo algumas manipulações algébricas, obtém-se um problema de programação quadrática similar àquele empregado para uma única SVM, na forma:

$$\max_{\alpha^{(t)}} Q(\alpha^{(t)}) = -\frac{1}{2} \sum_{t,l=1}^{N} y^{(t)} y^{(l)} \left(\varphi(\mathbf{x}^{(t)}) \cdot \varphi(\mathbf{x}^{(l)})\right) \alpha^{(t)} \alpha^{(l)} + \sum_{t=1}^{N} \alpha^{(t)}, \tag{6.35}$$

sujeito a

$$\begin{cases} \sum_{t=1}^{N} \alpha^{(t)} y^{(t)} = 0 \\ 0 \le \alpha^{(t)} \le C,\ t = 1,\ldots,N \end{cases}$$

onde $\alpha^{(t)}$ correspondem a multiplicadores de Lagrange.

Logo, a *l*-esima saída do classificador *j* pode ser expressa da seguinte forma:

$$s_{jl}(\mathbf{x}) = \sum_{t=1}^{N} \alpha_l^{(t)} \bar{y}_l^{(t)} K(\mathbf{x}, \mathbf{x}^{(t)}) + b_l \ , \tag{6.36}$$

Todo o equacionamento obtido foi baseado na função ε-insensível (veja equação 6.34), o mesmo pode ser realizado utilizando a função de perda ε-quadrática e a função de Huber (ambas definidas no Capítulo 4).

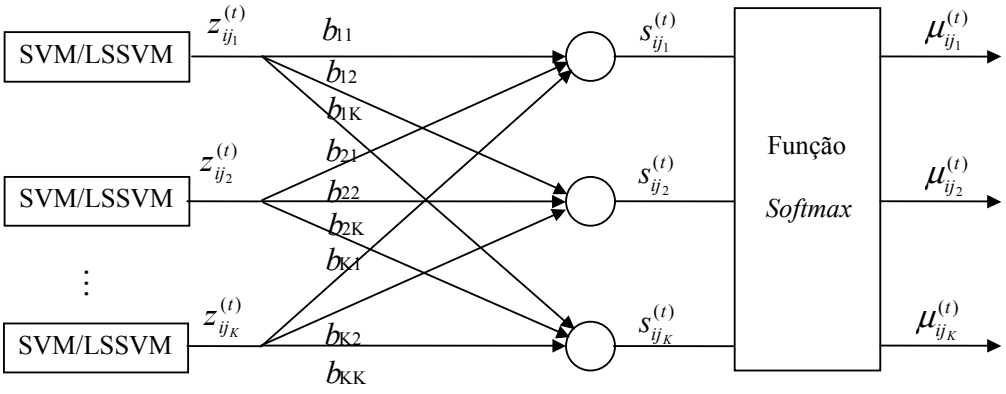


Figura 6.33 – Diagrama de blocos de um especialista *j* utilizando densidade condicional multinomial.

6.6.2 Caso 2: LS-SVM

Uma outra formulação equivalente para o problema definido em (6.30) consiste em introduzir um conjunto de variáveis adicionais, diferentes daquelas introduzidas na seção anterior, como segue:

$$\min_{\mathbf{w}_k, b_k, \xi_k} \ J(\mathbf{w}_k, b_k, \xi_k^{(t)}) = \sum_{t=1}^{N} h_j^{(t)} \sum_{i=1}^{K} \left(e_i^{(t)}\right)^2 \ , \tag{6.37}$$

$$\text{sujeito a } \begin{cases} \bar{y}_1^{(t)}[\mathbf{w}_1^T \varphi(\mathbf{x}^{(t)}) + b_1] = e_1^{(t)}, t = 1,\ldots,N \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \bar{y}_1^{(t)}[\mathbf{w}_1^T \varphi(\mathbf{x}^{(t)}) + b_1] = e_1^{(t)}, t = 1,\ldots,N \end{cases}. \tag{6.38}$$

Da mesma forma que na seção anterior, pode-se adicionar um termo de regularização para tornar o problema bem definido. Logo, a função-objetivo que será minimizada pode ser definida como segue:

$$\min_{\mathbf{w}_k, b_k, \xi_k} J(\mathbf{w}_k, b_k, \xi_k^{(t)}) = \frac{1}{2}\sum_{i=1}^{K} \mathbf{w}_k^T \mathbf{w}_k + C\sum_{t=1}^{N} h_j^{(t)} \sum_{i=1}^{K} \xi_i^{(t)} \tag{6.39}$$

Levando o problema para o espaço dual e fazendo algumas manipulações algébricas, obtém-se o seguinte sistema de equações lineares:

$$\begin{bmatrix} 0 & -\mathbf{Y}^T \\ \mathbf{Y} & \boldsymbol{\Omega} + \mathbf{V} \end{bmatrix} \begin{bmatrix} \mathbf{b} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \vec{\mathbf{1}} \end{bmatrix}, \tag{6.40}$$

onde $\mathbf{Y} = bloco\ diag\left\{\begin{bmatrix} \bar{y}_1^{(1)} & \ldots & \bar{y}_1^{(N)} \end{bmatrix}^T, \ldots, \begin{bmatrix} \bar{y}_K^{(1)} & \ldots & \bar{y}_K^{(N)} \end{bmatrix}^T\right\}$, $\boldsymbol{\Omega} = blocodiag\{\Omega_{(1)}, \ldots, \Omega_{(K)}\}$, $K(\mathbf{x}^{(k)}, \mathbf{x}^{(l)}) = \left(\varphi(\mathbf{x}^{(k)}) \cdot \varphi(\mathbf{x}^{(l)})\right)$, $\Omega_i^{(kl)} = \bar{y}_i^{(k)} \bar{y}_i^{(l)} K_i(\mathbf{x}^{(k)}, \mathbf{x}^{(l)}) + \mathbf{V}_i^{(kl)}\mathbf{I}$, $\mathbf{V} = blocodiag[\mathbf{V}_1; \mathbf{V}_2; \ldots; \mathbf{V}_K]$, $\mathbf{V}_j = diag\left\{1/(C.h_j^{(1)}), \ldots, 1/(C.h_j^{(N)})\right\}$, e o vetor solução $\mathbf{b} = [b_1; \ldots; b_K]$; $\boldsymbol{\alpha} = [\alpha_1^{(1)}; \ldots; \alpha_1^{(N)}; \alpha_2^{(1)}; \ldots; \alpha_2^{(N)}; \alpha_K^{(1)} \ldots \alpha_K^{(N)}]$. Repare que $\alpha_k^{(t)}$ representam os multiplicadores de Lagrange para a *k*-ésima saída do especialista *j*, referente ao padrão *t*.

Logo, a *l*-esima saída do classificador *j* pode ser expressa da seguinte forma:

$$f_{jl}(\mathbf{x}) = sign\left[\sum_{t=1}^{N} \alpha_l^{(t)} \bar{y}_l^{(t)} K(\mathbf{x}, \mathbf{x}^{(t)}) + b\right],$$

A solução para o sistema linear apresentado na equação (6.40) é equivalente à solução encontrada por um único LS-SVM (veja Capítulo 4), validando a proposta de modelagem do treinamento de cada especialista como um sistema linear de equações.

6.6.3 Algoritmo Proposto

O seguinte pseudo-código foi utilizado para treinamento da arquitetura de MEs para problemas de classificação, tendo SVM ou LS-SVM como especialistas.

1. Inicialização da *gating*;
   1.1. Inicialize os parâmetros da *gating* no intervalo [−0,2; 0,2];
2. Inicialização dos especialistas;
   2.1. Use o *K*-means ou um processo randômico para dividir os dados em *M* partições
   2.2. Atribua cada especialista a um das regiões;
3. Calcule a função de verossimilhança (veja Capítulo 3);
4. Enquanto a variação na função de verossimilhança é menor que um dado valor (nesta tese será utilizado 0,01)
   4.1. Passo E: Para cada par $(\mathbf{x}^{(t)}, y^{(t)})$, calcule a probabilidade a posteriori $h_j^{(t)}$ para cada especialista *j*, usando os valores atuais;
   4.2. Passo M:
      4.2.1. Para cada especialista *j*, solucione o problema definido em (6.35) ou em (6.40);
      4.2.2. Para a *gating*, solucione o problema de otimização definido no Capítulo 3, usando o método do gradiente;
5. Atualize os parâmetros da *gating* e do especialista e volte ao passo 4.

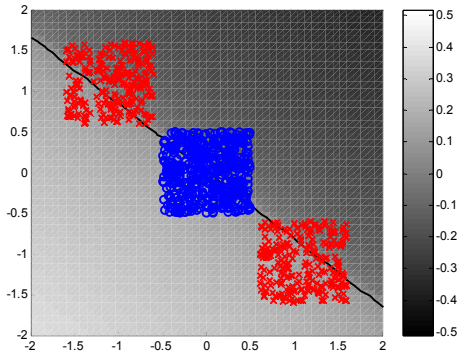
6.7 Experimentos computacionais para classificação

Nesta seção, serão apresentados os resultados obtidos com a abordagem proposta comparada com um único modelo. Para facilitar, esta seção será dividida em duas subseções, sendo que na primeira subseção apresentaremos resultados sobre conjuntos de dados artificiais, visando apenas mostrar o comportamento da ferramenta. Na segunda subseção, abordaremos problemas reais, visando comparar o desempenho da ferramenta proposta frente à utilização de um único modelo.

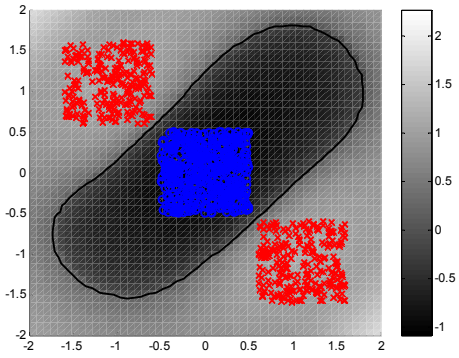
6.7.1 Experimento no. 1

Nesta primeira série de experimentos, a arquitetura ME-SVMs foi testada sobre um problema artificial simples já considerado por COLLOBERT *et al.* (2002), para o qual foram geradas 900 amostras de treinamento, 100 amostras de validação e 10000 amostras de teste. O problema tem duas classes não-linearmente separáveis (os dois quadrados externos na figura contêm amostras de uma única classe), com dois atributos como entrada. A Figura

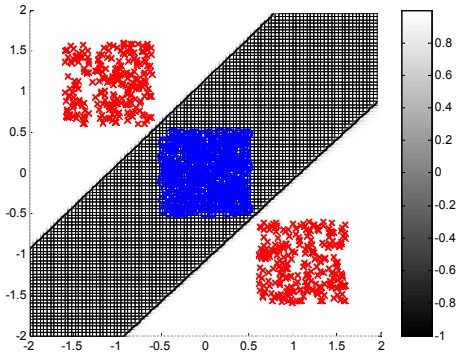
6.34 mostra a superfície de decisão obtida para uma LS-SVM com *kernel* linear, para uma LS-SVM com *kernel* gaussiano, e, finalmente, para uma MEs com dois especialistas LS-SVM com *kernel* linear. Observe que a mistura é capaz de combinar modelos muito simples, os quais não podem solucionar o problema isoladamente (veja Figura 6.34(a)), a fim de produzir uma superfície de decisão não-linear (veja Figura 6.34(c)).



(a) LS-SVM com *kernel* linear



(b) LS-SVM com *kernel* gaussiano



(c) Mistura de dois LS-SVM com *kernel* linear

Figura 6.34 – Comparação da superfície de decisão para (a) LS-SVM com *kernel* linear; (b) LS-SVM com *kernel* gaussiano; e (c) mistura de dois LS-SVM com *kernel* linear

6.7.2 Experimento no. 2

Neste experimento, será verificada a robustez da abordagem proposta em relação à variação dos parâmetros do *kernel*. Toda a análise realizada será restrita ao *kernel* RBF, isto é, à variação na abertura da gaussiana (σ), mantendo o valor de *C* fixo em 100. Resultados similares podem ser obtidos com outros *kernels*. O intervalo de busca dos parâmetros foi fixado entre $[2^{-10},2^{+15}]$. Para cada valor escolhido neste intervalo, foi realizada uma validação cruzada de forma a medir o desempenho do modelo. Com este objetivo, vários conjuntos de dados foram obtidos do repositório de aprendizado de máquina da Universidade de Wisconsin e do UCI (MURPHY & AHA, 1994). A Tabela 6.5 apresenta uma descrição dos conjuntos de dados utilizados.

Conjunto de Dados	Nº de Amostras	Nº Classe	Atributos		MEs	
			Contínuo	Discreto	Entradas	Saídas
Câncer wdbc[1]	569	2	30	---	30	2
Câncer colon	699	2	9	---	9	2
Diabetes	768	2	9	---	9	2

Tabela 6.5 – Descrição dos conjuntos de dados.

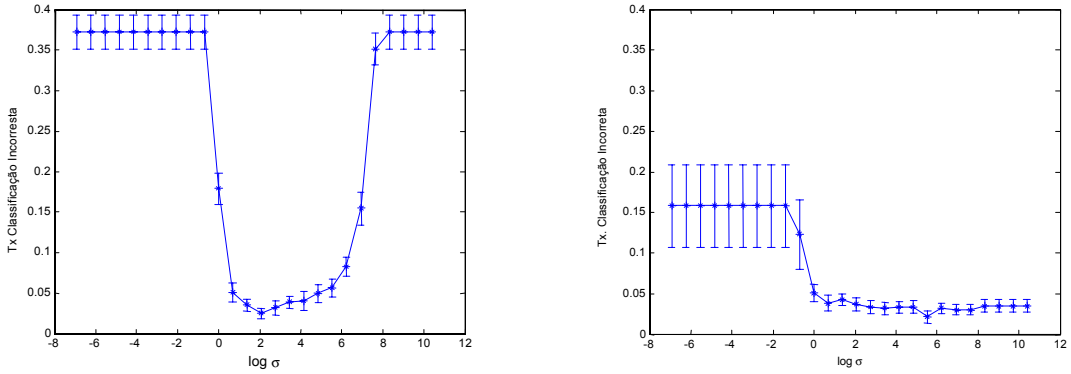
Os melhores resultados obtidos para cada conjunto de dados são apresentados nas Tabelas 6.6, 6.7 e 6.8 , com o valor associado ao melhor parâmetro da largura do *kernel*. As Figuras 6.35, 6.36 e 6.37 apresentam os resultados entre LS-SVM e ME[SVM] implementado via LS-SVM (resultados semelhantes podem ser obtidos utilizando o ME[SVM] implementado via SVM tradicional, para as 3 funções de perda) para cada valor do parâmetro do *kernel*. Conforme pode ser observado, a abordagem é menos sensível à variação no *kernel*, quando comparado com uma única SVM ou LS-SVM. No entanto, para os problemas estudados, o resultado obtido via MEs e por um único *kernel*, quando seus parâmetros são apropriadamente definidos, são equivalentes. Vale ressaltar que o custo

[1] Obtido a partir de ftp://ftp.cs.wisc.edu/math-prog/cpo-dataset/machine-learn/cancer/WDBC/

computacional para obtenção dos melhores parâmetros para o *kernel* é bem maior que o obtido via MEs, pois requer uma busca exaustiva.

Abordagens	Parâmetros					Melhor Resultado
	M	*h*	*C*	σ	Função de perda	Tx. Classificação Incorreta
SVM	---	---	100	32	ε - insensível	0,021053±0,007758
SVM	---		100	8	ε - quadrática	0,017544±0,004529
SVM	---		100	32	Huber	0,022807±0,007420
LS-SVM	2	---	100	8	-----	0,024593±0,006508
ME[SVM]	2	5	100	32	ε - insensível	0,021084±0,007302
	3	5	100	32		0,021065±0,006534
ME[SVM]	2	5	100	8	ε - quadrática	0,017564±0,004254
	3	5	100	8		**0,017504±0,004056**
ME[SVM]	2	5	100	64	Huber	0,022685±0,007254
	3	5	100	64		0,022765±0,007545
ME[LS-SVM]	2	5	100	256	------	0,021115±0,007771
	3	5	100	256	-----	0,021095±0,007264

Tabela 6.6 – Resultados comparativos para as diversas abordagens utilizadas para o problema do câncer wpdb.



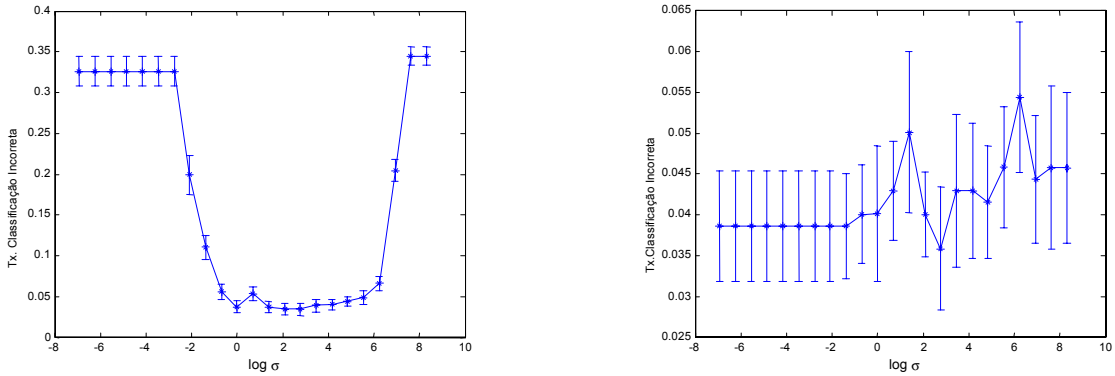
(a) LS-SVM

(b) Mistura de especialistas baseada em LS-SVM com $M = 2$ (dois especialistas)

Figura 6.35 – Resultado obtido para o câncer wdbc.

Abordagens	Parâmetros					Melhor Resultado
	M	*h*	*C*	σ	Função de perda	Tx. Classificação Incorreta
SVM	---	---	100	32	ε - insensível	0,0300410±0,00837
SVM	---		100	32	ε - quadrática	0,031470±0,008463
SVM	---		100	32	Huber	0,028613±0,008518
LS-SVM	2	---	100	16	-----	0,034327±0,007434
ME[SVM]	2	5	100	32	ε - insensível	0,030041±0,008370
	3	5	100	32		0,028894±0,007371
ME[SVM]	2	5	100	16	ε - quadrática	0,030170±0,009455
	3	5	100	32		0,028658±0,005421
ME[SVM]	2	5	100	32	Huber	**0,028613±0,008518**
	3	5	100	32		0,028613±0,008518
ME[LS-SVM]	2	5	100	16	------	0,035839±0,007546
	3	5	100	16	-----	0,030539±0,006146

Tabela 6.7 – Melhores resultados obtidos para o câncer de cólon.



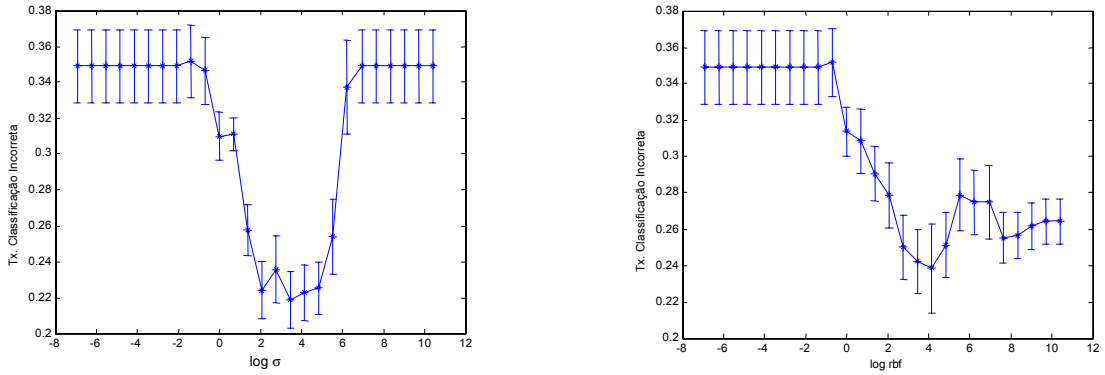
(a) LS-SVM

(b) Mistura de especialistas baseada em LS-SVM com $M = 2$ (dois especialistas)

Figura 6.36 – Resultado obtido para o câncer de cólon.

Abordagens	Parâmetros					Melhor Resultado
	M	*h*	*C*	σ	Função de perda	Tx. Classificação Incorreta
SVM	---	---	100	64	ε - insensível	0,224016±0,013913
SVM	---		100	32	ε - quadrática	0,222719±0,016598
SVM	---		100	64	Huber	0,224018±0,014699
LS-SVM	2	---	100	32	-----	0,218807±0,015920
ME[SVM]	2	5	100	32	ε - insensível	0,219561±0,012546
	3	5	100	32		0219150±0,015690
ME[SVM]	2	5	100	64	ε - quadrática	0,,219859±0,013546
	3	5	100	32		0,221226±0,014895
ME[SVM]	2	5	100	32	Huber	0,218905±0,012456
	3	5	100	32		0,221019±0,015679
ME[LS-SVM]	2	5	100	32	------	0,222542±0,018784
	3	5	100	64	-----	**0,218501±0,013896**

Tabela 6.8 – Melhores resultados obtidos para diabetes.



(a) LS-SVM

(b) Mistura de especialistas baseada em LS-SVM com $M = 2$ (dois especialistas)

Figura 6.37 - Resultado obtido para o diabetes.

6.7.3 Experimento no. 3

O objetivo deste experimento é demonstrar a viabilidade da abordagem proposta para problemas de classificação com múltiplas classes. Desta forma, foram geradas randomicamente nove distribuições gaussianas com centros localizados em regiões diferentes, sendo três distribuições para cada classe, conforme mostram as Figuras 6.38 e 6.39. Cada distribuição tem 70 amostras, totalizando 210 amostras para cada classe, ou seja, um conjunto de treinamento com 630 amostras (Figura 6.38) e um conjunto de teste com 1800 amostras (Figura 6.39).

A arquitetura da MEs tem dois especialistas, sendo que cada especialista é composto por 3 SVMs com *kernel* RBF, sendo que todos tinham variância unitária, parâmetro C fixo em 10^4 e função de perda ε-insensível. A Figura 6.40 mostra a evolução da função de verossimilhança para MEs com densidade de probabilidade multinomial.

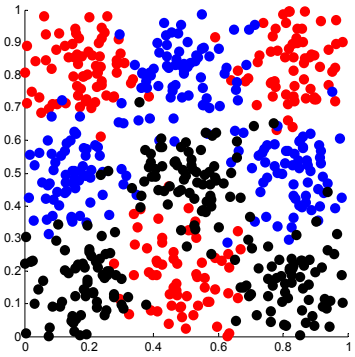


Figura 6.38 – Distribuição das classes para o conjunto de treinamento

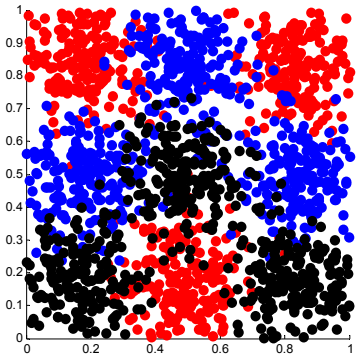


Figura 6.39 – Distribuição das classes para o conjunto de tese

A Figura 6.41 apresenta a fronteira de decisão gerada, destacando as amostras de treinamento para cada classe. Uma visão da distribuição das classes geradas pela arquitetura ME é apresentada na Figura 6.42. Inicialmente, é apresentada a distribuição das 3 classes envolvidas. Posteriormente, é apresentada a distribuição a posteriori entre uma classe e todas as outras.

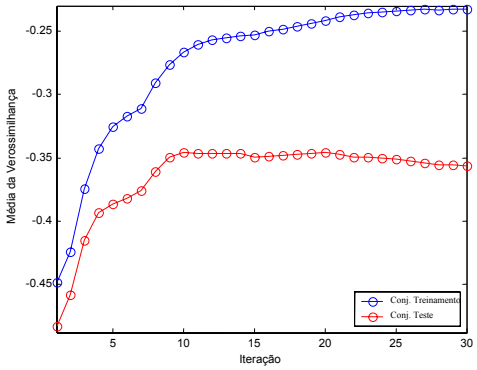


Figura 6.40 – Evolução da verossimilhança dos conjuntos de treinamento e teste

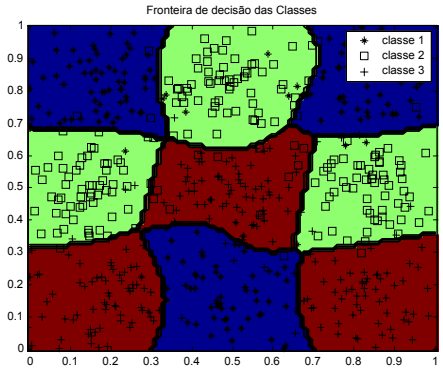


Figura 6.41 – Fronteira de decisão para o conjunto de treinamento

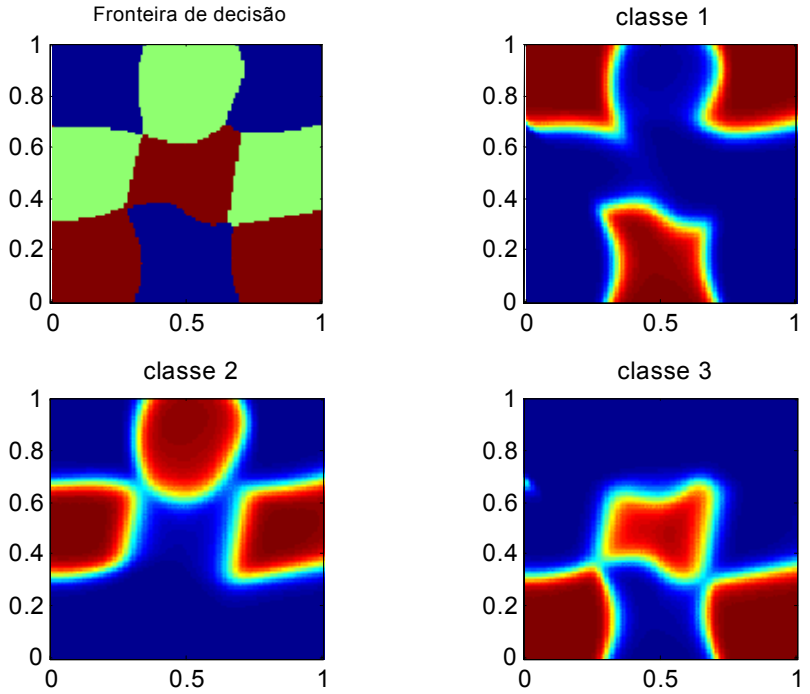


Figura 6.42 – Distribuição de probabilidade a posteriori para cada classe

Na Figura 6.43, é apresentada a distribuição das amostras entre os especialista. Isto pode ser realizado analisando o valor de h_i para cada especialista i. Duas estratégia foram utilizadas: uma atribuindo a amostra ao especialista com maior h_i e a outra abordagem

envolvendo simplesmente somar o valor de h_i para todas as amostras de treinamento. Para ambas, o valor total é dividido pelo número de amostras de treinamento. Como pode ser observado, as duas abordagens produziram aproximadamente o mesmo resultado, pois os valores de h_i estão próximos do valor unitário, ou seja uma separação abrupta entre os especialistas.

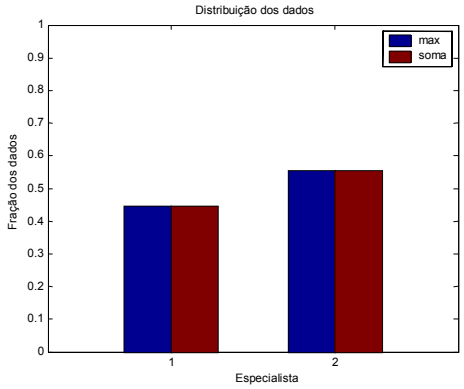


Figura 6.43 – Distribuição dos dados entre os especialistas

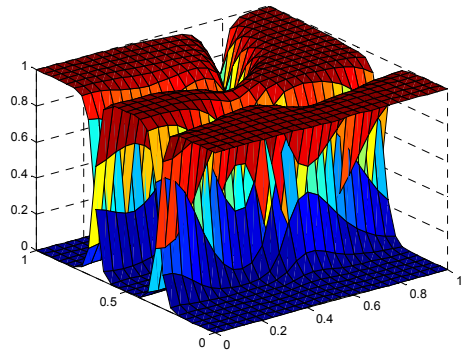


Figura 6.44 – Saída da *gating*, apresentando a área de atuação de cada especialista

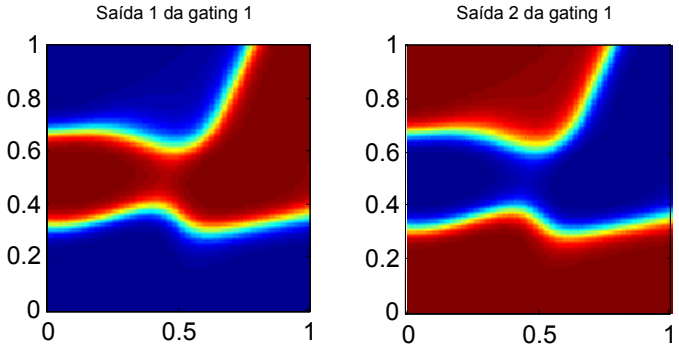


Figura 6.45 – Distribuição da saída da *gating*

A Figura 6.44 mostra a saída da rede *gating*. Há regiões em que ocorre sobreposição de atuação entre os especialistas. A distribuição da saída da *gating* é apresentado na Figura 6.45. Esta figura apresenta uma distribuição do espaço de entrada sobre uma perspectiva bi-dimensional.

6.8 Considerações Finais

Conforme já mencionado em capítulos anteriores, existem aspectos estruturais e paramétricos de projeto da abordagem SVM que podem conduzir a uma degradação de desempenho. Dentre tais aspectos estruturais, pode-se citar o tipo de função *kernel* mais apropriado para um determinado problema. Devido à existência de funções *kernel* com melhor desempenho nesta ou naquela região do espaço de entrada, a abordagem de mistura se mostra promissora no sentido de indicar de forma automática junto a qual região do espaço de entrada cada SVM deve se concentrar e qual deve ser a parametrização associada a cada especialista. Com base nisto, neste capítulo foi apresentada uma formulação matemática para treinamento de uma mistura de SVMs, a qual conduz a um problema quadrático que é similar àquele empregado no treinamento de uma única SVM (formulado no Capítulo 4), com nenhum aumento na dimensionalidade do problema quadrático associado. A formulação para LS-SVM também foi apresentada. Alguns experimentos computacionais são apresentados envolvendo classificação e regressão. Os resultados são promissores e incluem análise de sensibilidade paramétrica e, em alguns casos, interpretação do resultado obtido em termos do papel desempenhado pela *gating* e pelos especialistas.

Capítulo 7

Conclusão e Perspectivas Futuras

7.1 O enfoque da pesquisa

As ferramentas computacionais propostas durante o desenvolvimento da pesquisa e apresentadas ao longo desta tese, dentre outras potencialidades, mostraram-se eficientes no tratamento de vários aspectos, a saber:

- promover a configuração automática e sintonia de máquinas de aprendizado baseadas na abordagem SVM com *kernels* distintos, envolvendo vários parâmetros de projeto e, assim, diminuindo a necessidade de interferência do usuário no projeto de aprendizado de máquina;
- atribuir cada *kernel* a regiões diferentes do espaço de entrada, devido à possibilidade de haver regiões do espaço de entrada onde alguns *kernels* apresentam melhor capacidade de generalização;
- aumentar o poder de generalização da abordagem SVM, uma vez que as implementações computacionais normalmente não reproduzem o desempenho teórico esperado.

7.2 Contribuições e resultados obtidos

As contribuições originais deste trabalho estão basicamente concentradas nos capítulos 5 e 6, sendo que os capítulos anteriores fornecem os subsídios necessários ao posicionamento e à justificativa para as iniciativas adotadas ao longo destes dois capítulos. No entanto, vale ressaltar que todo o desenvolvimento apresentado ao longo dos capítulos 2, 3 e 4 foi concebido a partir da compilação e reposicionamento de diversos resultados já propostos na literatura, mas que geralmente são apresentados de forma isolada e com notação e abordagens diversas. Alguns dos temas levantados nestes capítulos são tópicos avançados de pesquisa em aprendizado de máquina e, por esta razão, não se encontram associados com conceitos mais elementares, numa linha coerente de argumentação. Portanto, a abrangência destes capítulos é inédita e foi motivada pela necessidade de especificar adequadamente, mas não exaustivamente, as reais potencialidades de modelos baseados em máquinas de vetores-suporte e em comitês de máquinas.

Os casos de estudo tomados ao longo do texto, em vários capítulos correspondem a problemas clássicos abordados na literatura e podem ser classificados em duas categorias: problemas triviais e problemas complexos, estes últimos caracterizados por não-linearidades, alta dimensão, baixa amostragem, chaveamento de contextos e/ou presença de ruído. Com relação aos problemas triviais, eles são importantes no contexto em que foram aplicados, no sentido de facilitar o entendimento de aspectos específicos da ferramenta.

As contribuições do Capítulo 5 envolvem as três fases associadas à abordagem ensemble: geração, seleção e combinação de modelos. Os principais tópicos abordados foram: (*i*) proposição de ensembles de SVMs heterogêneas; (*ii*) proposição de novos critérios de ordenação para seleção de componentes baseados em dimensão VC e margem de separação; (*iii*) proposição de critérios de seleção de componentes baseados em algoritmos genéticos; e (*iv*) extensão do conceito de ensemble de múltiplos estágios para SVM. No entanto, neste mesmo capítulo não foram exploradas as extensões propostas para os vários tipos de combinação existentes para ensemble. Apesar de atualmente já existirem propostas de extensão de SVM para ensembles, tais propostas baseiam-se principalmente na alteração do conjunto de dados de treinamento, utilizando alguma estratégia como *bagging* e *boosting*. Os resultados apresentados para as abordagens propostas foram

comparados com estratégias alternativas, produzindo melhor desempenho para os problemas estudados.

No Capítulo 6, por sua vez, é apresentada uma formulação original que permite a incorporação de máquinas de vetores-suporte como módulos de arquiteturas de mistura, levando a uma formulação matemática unificada. O resultado é uma mistura de especialistas baseada em máquinas de vetores-suporte, denotada ME[SVM]. O problema quadrático resultante é similar àquele associado a uma única máquina de vetores-suporte, mantendo a mesma dimensão do problema de programação quadrática (QP). A abordagem ME[SVM], além de permitir o uso de diferentes especialistas em regiões diferentes do espaço de entrada, também suporta combinação de diferentes *kernels*, tais como *kernels* polinomiais e funções de base radial. Os resultados de simulação a serem apresentados indicam que esta nova abordagem conduz a ganhos de desempenho em termos de generalização, quando comparada com uma única máquina de vetores-suporte. Devido à grande possibilidade de variar os parâmetros estruturais e paramétricos da SVM e da arquitetura ME, os parâmetros apresentados ou foram utilizados de modo a facilitar a compreensão, ou foram escolhidos de acordo com abordagens similares publicadas na literatura, ou correspondem àqueles nos quais a abordagem tradicional (empregando uma única SVM) apresentou melhor desempenho.

Ainda com relação aos Capítulos 5 e 6, os resultados apresentados são promissores, mas não podem ser interpretados como conclusivos, inclusive pelo fato de que os problemas de aplicação foram escolhidos arbitrariamente. O propósito básico foi sempre o de apontar encaminhamentos para a solução dos problemas relacionados com a abordagem SVM, visando ganho de desempenho e ampliação das possibilidades de aplicação.

7.3 Perspectivas futuras

As perspectivas futuras incluem tanto aqueles tópicos que não foram devidamente abordados ao longo da pesquisa, pelas mais variadas razões, quanto tópicos que ganham importância a partir dos resultados da própria pesquisa. São elas:

- espera-se que as novas perspectivas de implementação propostas ao longo dos capítulos 5 e 6 contribuam no sentido de viabilizar a implementação de uma nova geração de algoritmos e modelos para aprendizado de máquina, dotados de maior autonomia e flexibilidade. Duas são as conseqüências esperadas: maior automatização de todo o processo de aprendizado e maior eficiência no uso dos recursos computacionais disponíveis.
- seguindo a mesma linha adotada nos Capítulos 2, 3 e 4, deve-se concentrar esforços na elaboração de documentos capazes de apontar de forma direta as principais distinções entre as abordagens, suas vantagens e desvantagens, mesmo que restritas a casos de estudo específicos. Um benefício imediato desta iniciativa está no fornecimento de mais subsídios para permitir uma discriminação eficiente dos diversos modelos já propostos.
- investigar o comportamento de misturas de especialistas para especialistas heterogêneos, concebidos com aspectos estruturais e/ou paramétricos diferentes (por exemplo, tipos de *kernel* diferentes) ou treinados por estratégias diferentes.
- ainda no contexto de misturas, utilizar abordagens construtivas para melhor determinação do número ótimo de especialistas, de acordo para o problema considerado.
- ainda no contexto de misturas, estudar novas técnicas de inicialização, tanto dos especialistas quanto da rede, de forma a contribuir junto ao processo de convergência do algoritmo.
- estender as diversas propostas de comitês de máquinas para outras variações de SVM, principalmente aquelas que possuem uma interpretação probabilística da saída.
- aplicar as diversas metodologias propostas a outros problemas de classificação e regressão.
- estender as diversas metodologias propostas a problemas de treinamento não-supervisionado.
- estender as diversas metodologias propostas a problemas de treinamento supervisionado por inferência transdutiva, a qual é indicada junto a problemas de classificação em que há um número muito grande de amostras não-rotuladas disponíveis, além das amostras rotuladas disponíveis para treinamento supervisionado.

Apêndice A

Descrição dos Conjuntos de dados

Resumo: Este apêndice contém uma descrição detalhada de alguns dos principais conjuntos de dados utilizados ao longo desta tese.

A.1 Problema de Regressão

Os conjuntos de dados investigados para o problema de regressão foram os seguintes:

Housing Boston

Contém informação coletada pelo serviço de censo dos Estados Unidos a respeito das habitações na área popular de Boston e foi originalmente publicado por HARRISON & RUBINFELD (1978). Este contém 506 amostras, 13 atributos e duas saídas a serem estimadas: **nox**, é o nível de óxido nitroso; **price** é o valor médio das casas. Os dados tanto de entrada quanto de saída foram normalizados no intervalo [−1,1]. Cada saída foi tratada independentemente.

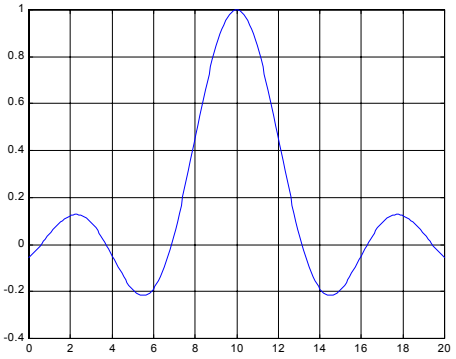
Dados Artificiais

Os conjuntos de dados artificiais gerados foram baseados nas cinco funções listadas na Tabela A.1. Na literatura, a função $g^{(1)}$: [0,20] $\rightarrow \Re$ foi empregada por VAPNIK (1995) para demonstrar a influência do número de vetores-suporte no erro de aproximação. As funções $g^{(2)}$, $g^{(3)}$, $g^{(4)}$ e $g^{(5)}$ ($g^{(k)}$: $[0,1]^2 \rightarrow \Re$, para $k = 2,...,5$), foram tratadas por HWANG *et al.* (1994) para comparar o desempenho de algoritmos construtivos em relação ao algoritmo de retro-propagação para perceptrons multicamadas (MLP). Somente para a função $g^{(1)}$ os dados de entrada foram normalizados no intervalo [−1,1].

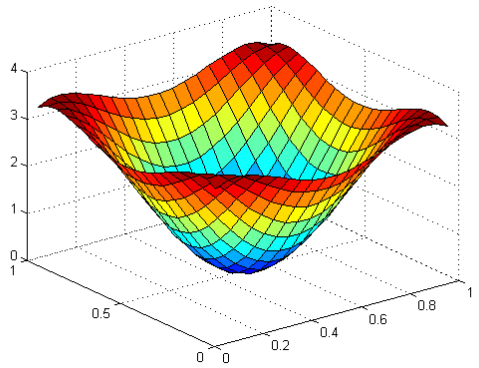
A motivação para se restringir a conjuntos de dados de baixa dimensionalidade é a possibilidade de uma visualização 3D das não-linearidades envolvidas. A Tabela A.1 apresenta a expressão para as cinco funções, e seu esboço gráfico é mostrado na Figura A.1.

Função Sinc	- Unidimensional $g^{(1)}(x_1) = \dfrac{\text{sen}(x_1 - 10)}{x_1 - 10}$ - Bi-dimensional $g^{(1)}(x_1, x_2) = \dfrac{\text{sen}\left(\sqrt{(x_1-10)^2 + (x_2-10)^2}\right)}{\sqrt{(x_1-10)^2 + (x_2-10)^2}}$
Função Radial	$g^{(2)}(x_1, x_2) = 24{,}234[r^2(0{,}75 - r^2)]$, com $r^2 = (x_1 - 0{,}5)^2 + (x_2 - 0{,}5)^2$
Função Harmônica	$g^{(3)}(x_1, x_2) = 42{,}659[0{,}1 + \bar{x}_1(0{,}05 + \bar{x}_1^4 - 10\bar{x}_1^2\bar{x}_2^2 + 5\bar{x}_2^4)]$, com $\bar{x}_i = x_i - 1/2 \quad i = 1,2$
Função Aditiva	$g^{(4)}(x_1, x_2) = 1{,}3356.[1{,}5(1 - x_1) +$ $e^{2x_1 - 1}\text{sen}(3\pi(x_1 - 0{,}6)^2) + e^{3x_2 - 1{,}5}\text{sen}(4\pi(x_2 - 0{,}9)^2)]$
Função complexa	$g^{(5)}(x_1, x_2) = 1{,}9[1{,}35 + e^{x_1 - x_2}\text{sen}(13(x_1 - 0{,}6)^2)\text{sen}(7x_2)]$

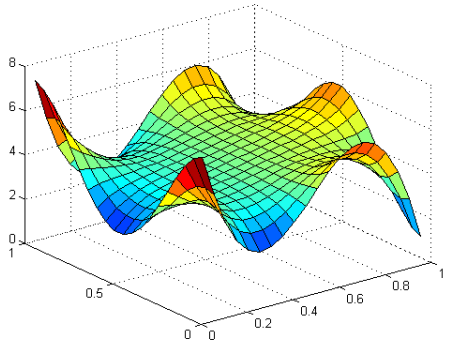
Tabela A.1- Funções para problemas de regressão



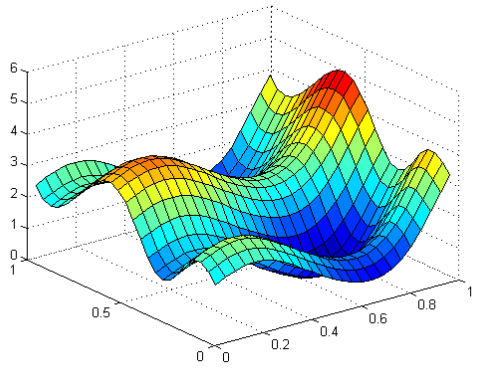
$g^{(1)}$



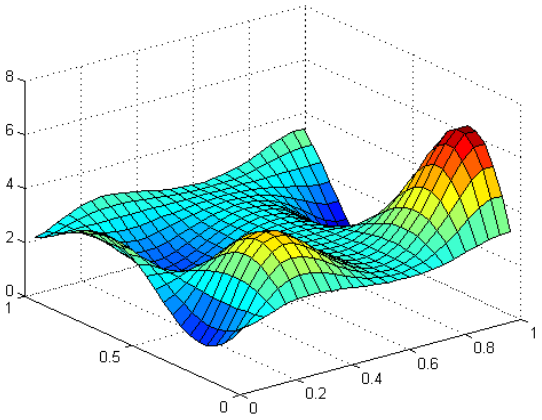
$g^{(2)}$



$g^{(3)}$



$g^{(4)}$



$g^{(5)}$

Figura A.1 Funções utilizadas em problema de regressão

No Capítulo 6, foram apresentados outros três problemas relacionados à regressão: predição de séries temporais, chaveamento entre processos e identificação de sistemas dinâmicos não-lineares. Como a apresentação realizada detalhou todos os aspectos relevantes de cada problema, estes não serão descritos nesta seção.

A.2 Problema de classificação

Os conjuntos de dados investigados para problemas de classificação foram os seguintes:

Duas espirais

O problema das duas espirais, primeiramente tratado por LANG & WITBROCK (1988) e descrito na Figura A.2, consiste de dois anéis (classes) entrelaçados, cujas equações são apresentadas abaixo:

Espiral #1	Espiral #2
$x = 1 + (r + 0{,}1) * \cos(t)$	$x = 1 - (r + 0{,}2) * \cos(t)$
$y = 1 + (r + 0{,}1) * \text{sen}(t)$	$y = 1 - (r + 0{,}2) * \text{sen}(t)$

onde r é definido no intervalo [0,1] e t é definido no intervalo $[0,3\pi]$. A idéia é categorizar os padrões de entrada como pertencente a uma das duas espirais (50% dos dados de treinamento para cada classe).

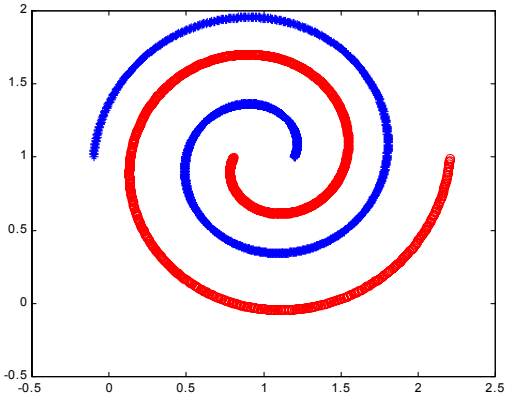


Figura A.2 – Problema das duas espirais.

Diabetes

Consiste em diagnosticar se o paciente mostra sinais de diabetes de acordo com critérios da Organização Mundial da Saúde. Este conjunto possui 768 amostras, com 8 atributos cada, sendo 500 amostras com diagnóstico negativo e 268 com diagnóstico positivo.

Câncer de Cólon (BCD)

Consiste em distinguir entre os pacientes com maior chance de desenvolver câncer de cólon e aqueles com menor chance, levando em conta dados histológicos produzidos pelas biópsias obtidas de ambos os grupos. O conjunto de dados consiste de 699 exemplos, cada um com 9 atributos. Dos 699 pacientes examinados, 458 são indicados como amostras de tumor benigno (65,5%) e 241 são indicados como tumores malignos (34,5%).

Câncer de Cólon (WPDC)

Este conjunto de dados é semelhante ao BCD, embora difira no número de amostras e no número de atributos. Este conjunto possui 569 amostras, cada uma com 30 atributos, sendo 357 pacientes com tumores classificados como benigno e 212 como maligno.

Hepatite

O conjunto de dados de hepatite está associado a um problema de classificação binária com 155 exemplos, sendo que a tarefa do classificador é indicar, de acordo com o perfil do paciente, se ele corre ou não risco de morte. Há 19 atributos de entrada e 167 valores faltantes, algo que aumenta a dificuldade do problema. Dos 155 pacientes, 32 correspondem aos pacientes que não correm risco de morte e 123 àqueles que podem morrer.

IRIS

O conjunto de dados da IRIS (FISHER, 1936) é um dos conjuntos de dados mais conhecidos encontrado na literatura de reconhecimento de padrões. O conjunto de dados contém 4 atributos e 3 espécies de gênero *Iris*, e o objetivo é classificar corretamente as amostras de cada uma das 3 espécies baseado nestes 4 atributos. Para visualizar a distribuição das amostras, é possível tomar as duas características que contêm a maior parte da informação a respeito da classe, a saber: comprimento da pétala e largura da pétala. São 50 amostras para cada classe.

Glass

Consiste da identificação de tipos de vidro, sendo motivado por investigação criminal. Na cena do crime, o vidro deixado pode ser usado como evidência, se for corretamente classificado. O conjunto contém 6 classes, onde as classes contêm as seguintes quantidades de amostras: 70, 76, 17, 13, 9, 29, totalizando 214 amostras.

Reconhecimento de Câncer baseado em Microarray

Este é um problema de identificação de células cancerígenas a partir de dados de expressão gênica. Este problema foi inicialmente investigado por ALON *et al.* (1999) e representa um grande desafio, uma vez que o número de atributos por padrão é muito alto e as amostras de treinamento são esparsas. O conjunto de dados consiste de 62 amostras de tecido, cada uma com 2000 atributos (níveis de expressão gênica). Dentre os 62 tecidos, 22 são considerados normais e 40 apresentam câncer. A idéia é discriminar entre dois grupos de pacientes,

pacientes doentes (classe 1) e pacientes normais (classe 2), baseado nos dados de expressão gênica do tecido do cólon.

Apêndice B

Publicações vinculadas à Tese

Resumo: Os artigos científicos listados a seguir estão diretamente vinculados à tese e foram produzidos ao longo do desenvolvimento do plano de trabalho vinculado ao Projeto de Pesquisa. Eles serão apresentados em ordem cronológica de publicação.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Fuzzy Systems Design via Ensembles of ANFIS. *Proceedings of the IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE'2002), vol. 1, pp. 506-511, in the 2002 IEEE World Congress on Computational Intelligence (WCCI'2002), Honolulu, Hawaii, May 12-17, 2002.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Ensembles of Support Vector Machines for Regression Problems. *Proceedings of the IEEE International Joint Conference on Neural Networks* (IJCNN'2002), vol. 3, pp. 2381-2386, in the 2002 IEEE World Congress on Computational Intelligence (WCCI'2002), Honolulu, Hawaii, May 12-17, 2002.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Mixture of Experts Applied to Nonlinear Dynamic Systems Identification: A Comparative Study. Proceedings of the VII Brazilian Symposium on Neural Networks, Porto de Galinhas, Recife, Novembro 11-14, pp. 162-167, 2002.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Model Selection based on VC-dimension for Heterogeneous Ensembles of Support Vector Machines. Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC2002), Nottingham, United Kingdom, pp. 459-464, December 2002.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. On Localized Mixture of Support Vector Machine Experts. Proceedings of the 4th International Conference on Recent Advances in Soft Computing (RASC2002), Nottingham, United Kingdom, pp. 465-470, December 2002.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Ensembles of Support Vector Machines for Classification Tasks with Reduced Training Sets, *WSEAS Transactions on Systems*, Issue 2, Volume 2, pp. 370-375, April 2003. (ISSN: 1109-2777)

Coelho, A.L.V., **Lima, C.A.M.**, Von Zuben, F.J. Hybrid Genetic Training of Gated Mixtures of Experts for Nonlinear Time Series Forecasting, Proceedings of the 2003 IEEE International Conference on Systems, Man & Cybernetics, October 5th-8th, Washington DC, USA, pp. 4625-4630, 2003.

Coelho, A.L.V., **Lima, C.A.M.**, Von Zuben, F.J. GA-based Selection of Components for Heterogeneous Ensembles of Support Vector Machines. 2003 Congress on Evolutionary Computation (CEC'2003), Canberra, Australia, 8^{th}-12^{th} December, vol. 3, pp. 2238-2245, 2003.

Lima, C.A.M., Coelho, A.L.V., Von Zuben, F.J. Embedding Support Vector Machines into Localised Mixtures of Experts. *in* Lofti, A. & Garibaldi, J.M. (eds.) Applications and Science in Soft Computing, Advances in Soft Computing Series, Springer, pp. 155-162, 2004. (ISBN: 3-540-40856-8)

Lima, C.A.M., Villanueva, W.J.P., dos Santos, E.P., Von Zuben, F.J. Mistura de Especialistas Aplicada à Predição de Séries Temporais Financeiras. Anais do VIII Simpósio Brasileiro de Redes Neurais (SBRN'2004), São Luís, MA, 29 de setembro a 1 de outubro, paper no. 3708, 2004.

Lima, C.A.M., Coelho, A.L.V., Villanueva, W.J.P., Von Zuben, F.J. Gated Mixtures of Least Squares Support Vector Machine Experts Applied to Classification Problems.

Proceedings of the 5th International Conference on Recent Advances in Soft Computing (RASC2004), Nottingham, United Kingdom, pp. 494-499, December 2004.

Lima, C.A.M., Villanueva, W.J.P., dos Santos, E.P., Von Zuben, F.J. A Multistage Ensemble of Support Vector Machine Variants. Proceedings of the 5th International Conference on Recent Advances in Soft Computing (RASC2004), Nottingham, United Kingdom, pp. 670-675, December 2004.

Villanueva, W.J.P., **Lima, C.A.M.**, dos Santos, E.P., Von Zuben, F.J. Mixture of Heterogeneous Experts Applied to Time Series: A Comparative Study. Proceedings of the International Joint Conference on Neural Networks (IJCNN'2005), Montreal, Quebec, Canada, July 31 - August 4, 2005. (to appear)

Índice Remissivo de Autores

Keerthi *et al.*, 1999 182
Keerthi *et al.*, 2001 182, 183
Keerthi & Gilbert, 2002 182
Kim *et al.,* 2002 206
Knerr, 1990 146, 203
Kohavi, 1995 197
Kohavi & John, 1997 197
Krebel, 1999 146
Kullback & Leibler, 1951 40
Kwok, 1998 204, 236, 250
Kwok, 2001 192
Lang & Witbrock, 1988 218
Langley & Simon, 1995 1
Laskov, 2000 182
LeBlanc & Tibshirani, 1993 32
Lee, 1997 33
Lee *et al*., 2001 232, 233
Lee & Srihari, 1993 235
Lima et al, 2002 11, 203, 205, 221, 230, 267
Lima et al, 2004 185, 239
Liu & Yao, 1999 231
Luenberguer, 1984 181
Mackay, 1997 90
Mackay, 2003 2
Maclin & Shavlik, 1995 12
Mangasarian, 1964, 1969 97
Mao, 1998 10
Mattera & Haykin, 1999 191, 192
McCullagh & Nelder, 1983 67
McLachlan & Basford, 1988 49, 50, 248

Referências Bibliográfica

Al-Ghoneim, K.; Kumar, B. V. K. V. Learning ranks with neural networks. *Applications and Science of Artificial Neural Networks: Proceedings of the SPIE*, vol. 2492, pp. 446-464, 1995.

Alon, U.; Barkai, N.; Notterman, D. A.; Gish, K.; Ybarra, S.; Mack, D.; Levine, A. J. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences of the United States of America*, vol. 96, no. 12, pp. 6745-6750, 1999.

Andrews, D. F.; Bichael, P. J.; Hampel, F. R.; Huber, P. J.; Rogers, W. H.; Tukey, J. W. **Robust estimate of location: survey and advances**, Princeton, NJ: Princeton University Press, 1972.

Bäck, T.; Fogel, D. B.; Michalewicz, editors. **Handbook of Evolutionary Computation**. Institute of Physics Publishing and Oxford University Press, 1997.

Bahler, D.; Navarro, L. Methods for Combining Heterogeneous Sets of Classifiers. 17th Natl. *17th Natl. Conf. on Artificial Intelligence (AAAI 2000)*, Workshop on New Research Problems for Machine Learning, 2000.

Baxt, W. G. Improving the accuracy of an artificial neural network using multiple differently trained networks. *Neural Computation*, vol. 4, no 5, pp.135-144, 1992.

Berry, M. J. A.; Linoff, G. **Data mining Techniques: For Marketing, Sales, and Customer Support**. New York, Wiley, 1997.

Bioch, J. C.; Meer, O. V. D.; Potharst, R. Classification using Bayesian neural nets. *IEEE International Conference on Neural Networks*, vol. 3, pp. 1488-1493, 1996.

Bishop, C. M. Improving the generalization properties of radial basis function neural network. *Neural Computation*, vol. 3, no 4, pp. 579-588, 1991.

Bishop, M. C. **Neural Networks for Pattern Recognition**. Oxford University, Press, Oxford, 1995.

Bishop, C. M.; Qazaz, C. S. Regression with input-dependent noise: A Bayesian treatment. In M. C. Mozer, M. I. Jordan, and T. Petsch, editors, *Advances in Neural Information Processing Systems*, vol. 9, Cambridge MA, MIT Press, 1997.

Bishop, C. M.; Tipping, M. E. A hierarchical latent variable model for data visualization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no 3, pp. 281-293, march, 1998.

Blake, C.; Merz, C. UCI Repository of Machine Learning Databases, 1998.

Boser, B. E.; Guyon, I. M.; Vapnik, V. N. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *5th Annual ACM Workshop on COLT*, pp. 144-152, Pittsburgh, PA, 1992, ACM Press.

Bradley, P. S.; Mangasarian, O. L. Feature selection via concave minimization and support vector machines. *Proceedings 13th Int. Conf. Machine Learning*, San Francisco, CA, pp. 82-90, 1998.

Bradley, P. S.; Mangasarian, O. L.; Street, W. N. Feature selection via mathematical programming. *INFORMS Journal Computation*, vol. 10, pp. 209-217, 1998.

Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J. Classification and Regression Trees. *Wadsworth International Group*, Belmont, CA, 1984.

Breiman, L. Stacked regression. *Technical Report 367*, Statistics Departament, University of California, Berkeley, 1993.

Breiman, L. Arcing classifiers. *Technical Report 460*, Statistics Departament, University of California, Berkeley, 1996a.

Breiman, L. Bagging predictors. *Machine Learning*, vol. 24, no 2, pp. 123-140, 1996b.

Breiman, L. Combining predictors. In **Combining artificial neural nets**, Sharkey, A. J. C., Ed., Springer, 31, 1999.

Breiman, L. Randomizing outputs to increase prediction accuracy. *Machine Learning*, vol. 40, 229, 2000.

Brown, M.; Grundy, W.; Lin, D.; Cristianini, N.; Sugnet, C.; Furey, T.; Ares, M.; Haussler, D. Knowledge-based analysis of microarray gene expression data using support vector machines. Technical report, University of California in Santa Cruz, 1999.

Burges, C. J. C. A tutorial on support vector machines for pattern recognition. *Data Mining Knowledge Discovery*, vol. 2, no 2, pp. 1-47, 1998.

Bottou, L.; Cortes, C.; Denker, J. S.; Drucker, H.; Guyon, I.; Jackel, L. D., LeCun, Y.; Muller, U. A.; Sackinger, E.; Simard, P.; Vapnik, V. Comparison of classifier methods: a case study in handwritten digit recognition, *Proceedings of the 12th IAPR International Conference on Pattern Recognition, Conference B: Computer Vision & Image Processing.*, (Jerusalem), pp. 77-82, Oct. 1994.

Campbell, C.; Cristianini, N.; Smola, A. Instance selection using support vector machines, *Machine Learning*, 1999.

Chapelle, O.; Vapnik, V. Model selection for support vector machines, *Advances in Neural Information Processing System*, vol. 12, ed. S. A. Solla, T. K. Leen and K.-R. Muller, MIT Press, 2000.

Cherkauer, K. J. Human expert level performance on a scientific image analysis task by a system using combined artificial neural networks. In P. Chan, S. Stolfo, D. Wolpert

(Eds), *Proceeding AAAI-96 Workshop on Integrating Multiple Learned Models for Improving and Scaling Machine Learning Algorithms*, Portland, OR, AAAI Pres, Menlo Park, CA, pp. 15-21, 1996.

Cherkassky, V.; Mulier, F. **Learning from data: Concepts, Theory, and Methods**, New York:Wiley, 1998.

Cherkassky, V. and Y. Ma. Selecting the loss function for robust linear regression. under review in Neural Computation, 2002.

Cherkassky, V.; Ma, Y. Practical selection of SVM parameters and noise estimation for SVM regression, *Neurocomputing* vol. 17, no 1, 113–126, 2004.

Collobert, R.; S. Bengio; Y. Bengio. A Parallel Mixture of SVMs for Very Large Scale Problems, *Neural Computation*, vol. 14, no 5, pp. 1105-1114, 2002.

Cortes, C.; Vapnik, V. Support vector networks. *Machine Learning*, vol. 20, pp. 273-297, 1995.

Cover, T.; Thomas, J. **Elements of Information Theory**. Willey Series in Communications, New York, 1991.

Cristianini, N.; Campbell, C.; Shawe-Taylor, J. Dynamically adapting kernels in support vector machines, Advances in *Neural Information Processing Systems*, vol 11, ed. M. Kearns, S. A. Solla, and D. Cohn, MIT Press, pp. 204-210, 1999.

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

Cristianini, N.; Schölkopf, B. Support Vector Machines and Kernel Methods The New Generation of Learning Machines, *AI Magazine*, vol 23, no 3, 2002.

Crowder, III R. S. Predicting the Mackey-Glass time series with cascade-correlation learning. In. D. Touretzky, G. Hinton, and T. Sejnowski, editors, *Proceedings of the Connectionist Models Summer School*, pp. 117-123, Carnegic Mellon University, 1990.

Cunningham, P.; Carney, J.; Jacob, S. Stability problems with artificial neural networks and the ensemble solution. *Artificial Intelligence in Medicine*, vol. 20, no 3, pp. 217-225, 2000.

DeCoste, D.; Schöolkopf, B. Training invariant support vector machines. *Machine Learning*, vol. 46, no 1-3, pp.161–190, 2002.

Dempster, A. P.; Laird, N. M.; Rubin, D. B. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, B, vol. 39, no 1, pp. 1-38, 1977.

Dietterich, T. G. Ensemble methods in machine learning, in: *Proceedings International Workshop on Multiple Classifier Systems (MCS)*, LNCS 1857, pp. 1-15, Italy, Springer, 2000.

Dietterich, T. G. An experimental comparison of three methods for construing ensembles of decision trees; bagging, boosting and randomization. *Machine Learning*, vol. 40, 139-157, 2000.

Dong, Jian-xiong; Krzyzak, A.; Suen, C. Y. A Fast SVM Training Algorithm. *International Journal of Pattern Recognition and Artificial Intelligence,* vol. 17, no 3, pp. 367-384, 2003.

Donoho, D. L.; Huber, P. J. The notion of breakdown point. *A Festschift for Erich Lehmann*, (Ed. P. Bickel, K. Doksum, J. L. Hodges Jr.), Belmont, CA: Wadsworth, 1983.

Drucker, H.; Shapire, R.; Simard, P. Improving performance in neural networks using a boosting algorithm. In S. J. Hanson, J. D. Cowan, C. L. Giles (Eds.), *Advances in Neural Information Processing System 5*, Denver, CO, Morgan Kaufamann, San Mateo, CA, pp. 42-49, 1993.

Drucker, H., Cortes, C., Jackel, L. L. D.; LeCun, Y.; Vapnik, V. Boosting and other ensemble methods. *Neural Computation*, vol. 6, no 6, pp. 1289-1301, 1994.

Drucker, H. Improving regressors using boosting techniques. *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 107-115. Morgan Kaufmann, 1997.

Drucker, H. Boosting neural networks. In **Combining artificial neural nets**, Sharkey, A. J. C., Ed.; Springer, 31, 1999.

Efron, E.; Tibshirani, R. **An introduction to the Bootstrap**. Chapman & Hall, New York, 1993.

Evgeniou, T.; Pontil, M.; Papageorgiou, C.; Poggio, T. Image representations for object detection using kernel classifiers. In *Asian Conference on Computer Vision*, 2000.

Fisher, R. A. The use of multiple measurements in taxonomic problems, *Annals of Eugenics* vol. **7**, pp. 179-188, 1936.

Flake G. W.; Lawrence, S. Efficient svm regression training with smo. *Machine Learning*, vol. 46, no 1-3, pp. 271–290, March, 2002.

Fogel, D. B. **Evolutionary Computation: Toward a New Philosophy of Machine Intelligence**, 2nd edition, The IEEE Press, 1999.

French, S. Group consensus probability distributions: a critical survey. *Bayesian Statistics*, vol. 2, pp. 183-202, 1985.

Freund, Y. Boosting a weak algorithm by majority. *Information and Computation*, vol. 121, no 2, pp. 256-286, 1995.

Freund, Y.; Shapire, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Proceedings EuroCOLT-94*, Barcelona, Spain, Springer-Verlag, Berlin, pp. 23-27, 1995.

Freund, Y.; Shapire, R. Experiments with a new boosting algorithm. *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 149-156. Morgan Kaufmann, 1996.

Friedman, J. H. Multivariate adaptive regression splines. *Annals Statistics*, vol. 19, pp. 1-41, 1991.

Friedman, J.; Hastie, T.; Tibshirani, R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors), *Annals of Statistics*, vol. 28, no. 2, pp. 337-407, 2000.

Friess, T. T.; Cristianini, N.; Campbell, C. The kernel adatron algorithm: a fast and simple learning procedure for support vector machine. *Proc. 15th International Conference on Machine Learning*, Morgan Kaufman Publishers, 1998.

Fritsh, J. **Modular neural networks for speech recognition**. Master' thesis, Carnegie Mellon University & University of Karlsrube, 1996.

Fritsch, J.; Finke, M.; Waibel, A. Context-dependent hybrid HME/HMM speech recognition using polyphone clustering decision trees. *Processing of ICASSP*-97, 1997.

Fritsch, J; Finke, M.; Waibel, A. Adaptively growing hierarchical mixtures of experts. *Advances in Neural Information Processing Systems 9.* Eds. M. C. Mozer, M. I. Jordan, Petsche, T. MIT Press, 55 Hayward St., Cambridge, MA, 02142-1399, pp. 459-465, 1997.

Furey, T. S.; Duffy, N.; Cristianini, N.; Bednarski, D.; Schummer, M.; Haussler, D. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, vol. 16, no 10, pp. 906-914, 2000.

Gelman, A.; Carlin, J. B.; Stern, H. S; Rubin, D. B. **Bayesian Data Analysis**, Chapman & Hall, New York, 1995.

Geman S., E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, vol. 4, pp. 1-58, 1992.

Girosi, F., Jones, M., Poggio, T. Regularization Theory and Neural Networks Architectures. *Neural Computation*, vol. 7, no. 2, pp. 219-269, 1995.

Girosi, F. An equivalence between sparse approximation and support vector machines. *Neural Computation*, vol. 10, no 6, pp. 1455-1480, 1998.

Golub, G. H; Van Loan, C. F. *Matriz Computations*, Baltimore MD: Johns Hopkins University Press, 1989.

Golub T.R., Slonim D.K., Tamayo P., Huard C., Gaasenbeek M., Mesirov J.P., Coller H., Loh M.L., Downing J.R., Caligiuri M.A., Bloomfield C.D., Lander E.S. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286(5439):531-7, 1999.

Guermeur, Y. Combining discriminant models with new multi-class SVMs. *Technical Report NeuroCOLT2, 2000-086*. 2000.

Gunn S. Support vector machine for classification and regression, *Technical Report ISIS-1-98, Image Speech & Intelligent Systems Group*, University of Southampton, 1998.

Gutta, S.; Wechsler, H. Face recognition using hybrid classifier systems. *Processing ICNN-96*, Washington, DC, *IEEE Computer Society Press*, Los Alamitos, CA, pp. 1017-1022, 1996.

Guyon, I; Boser, B.; Vapnik, V. Automatic capacity tuning of very large VC-dimension classifiers. In Stephen Jose Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Advances in Neural Information Systems*, vol. 5, pp. 147-155, Morgan Kaufmann, San Mateo, CA, 1993.

Guyon, I.; Weston, J.; Barnhill, S.; Vapnik, V. Gene selection for cancer classification using vector machines. *Machine Learning*, vol. 46, no 1-3, pp. 389-422, 2002.

Hagan, M.; Menhaj, M. Training Feedforward Networks with the Marquardt Algorithm. *IEEE Transactions on Neural Networks*, vol. 5, no 6, pp. 989-993, November 1994.

Hampel, F. R.; Ronchetti, E. M.; Rousseuw, P. J.; Stahel, W. A. **Robust statistics: the approach based on inference functions**. John Wiley & Sons, New York, 1986.

Hampshire, J.; Waibel, A. A novel objective function for improved phoneme recognition using time-delay neural networks. *IEEE Transactions on Neural Networks*, vol. 1, no 2, pp. 216-228, 1990.

Hansen, L. K.; Salamon, P. Neural network ensembles, *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 12, no 10, pp. 993-1001, 1990.

Hansen, L. K.; Liisberg, L.; Salamon, P. Ensemble methods for handwritten digit recognition. *Processing IEEE Workshop on Neural Networks for Signal Processing, Helsingoer*, Denmark, IEEE Processing Piscataway, NJ, pp. 333-342, 1992.

Hashem, S. and B. Schmeiser. Improving model accuracy using optimal linear combinations of trained neural networks, *IEEE Transactions on Neural Networks*, vol. 6, no 3, pp. 792-794, 1995.

Hashem S. Optimal linear combinations of neural networks. *Neural Networks*, vol. 10, no 4, pp. 599-614, 1997.

Hassem, S. **Optimal Linear Combinations of Neural Networks**. PhD thesis, School of Industrial Engineering, Purdue University, 1993.

Hassem, S. Effects of collinearity on combining neural networks. Connection Science: Special Issue on Combining Artifical Neural Networks: Ensemble Approaches, vol. 8, no 3 & 4, pp. 315-336, 1996.

Hassem, S. Optimal Linear Combinations of Neural Networks. *Neural Networks*, vol. 10, no 4, pp. 599-614, 1997.

Haykin, S. **Neural networks–A comprehensive foundation**, 2nd. Ed., Prentice Hall, 1999.

Hong, S. G.; Oh, S. K., Kim, M. S.; Lee, J. J. Nonlinear time series modeling and prediction using Gaussian RBF network with evolutionary structure optimization. *Electronics Letters*, vol. 37, no 10, pp. 639-640, 2001.

Hornik, K.; Stinchombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural Networks*, no 2, pp.359-366, 1989.

Hsu, C.-W.; Lin, C.-J. A comparison on methods for multi-class support vector machines , IEEE Transactions on Neural Networks, vol. 13, pp. 415-425, 2002.

Huang, F. J.; Zhou, Z. –H.; Zhang, H. –J, Chen, T. H. Pose invariant face recognition. *Processing 4th IEEE International Conference on Automatic Face and Gesture Recongnition*, Grenoble, France, *IEEE Computer Society Press*, Los Alamitos, CA, pp. 245-250, 2000.

Huber, P. J. **Robust statistics**. New York, John Wiley and Sons Inc., 1981.

Hunag, Y. S.; Suen, C. Y. A method of combining multiple experts for the recognition of unconstrained handwritten numerals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no 1, pp. 90-94, 1995.

Hwang J-N.; Lay, S.; Maechler, M.; Martin, R.; Schimert, J. Regression modeling in back-propagation and project pursuit learning, *IEEE Transactions on Neural Networks* vol. 5, no 3, 342-353, 1994.

Inoue, H.; Narihisa, H. Predicting Chaotic Time Series by Ensemble Self-Generating, *International Joint Conference on Neural Networks IJCNN*, vol. 2, pp. 231-236, 2000.

MacKay, D. J. C. **Information Theory, Inference & Learning Algorithms**, Cambridge University Press, 2003.

Jaakkola, T. S.; Haussler, D. Exploiting generative models in discriminative classifier. In M. S. Kearns, S. A Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing System*, vol. 11, MIT Press, 1998.

Jaakola, T.; Haussler, D. Probabilistic kernel regression models. *Proceedings of the 1999 Conference on AI and Statistics*, 1999.

Jacobs, R. A.; Jordan, M. I.; Nowlan, S. J.; Hinton, G. E. Adaptive mixtures of local experts. *Neural Computation*, vol. 3, no 1, pp. 79-87, 1991.

Jacobs, R. A. Methods for combining experts' probability assessments. *Neural Computation*, vol. 7, pp. 867-888, 1995.

Jain, A. K.; Duin, R. P. W.; Mao, J. Statistical pattern recognition: A review, *IEEE Transactions Pattern Anal. Machine Intelligent*, vol. 22, pp. 4-37, Jan. 2000

Jang, J. S. ANFIS: Adaptive-network-based fuzzy inference system, *IEEE Transactions on System, Man, and Cybernetics*, vol. 23, no 3, pp. 665-685, May/June, 1993.

Jimenez, D. Dynamically weighted ensemble neural networks for classification. *International Joint Conference on Neural Networks IJCNN*, vol. 1, Anchorage, AK, *IEEE Computer Society Press*, Los Alamitos, CA, pp. 753-756, 1998.

Joachims, T. Making large-scale support vector machine learning practical. In B. Sch¨olkopf, C. Burges, and A. Smola, editors, *Advances in kernel methods: Support Vector Machines*. MIT Press, Cambridge, MA, December 1998.

Jones, R. D.; Lee, Y. C.; Barnes, C. W.; Flake, G. W.; Lee, K.; Lewis, P. S. Function approximation and time series prediction with neural networks. In. *International Joint Conference on Neural Networks IJCNN*, pp. 649-665, 1990.

Jordan, M. I; Jacobs, R. A. Hierachies of adaptive experts. *Advances in Neural Information Processing Systems 4*, J. Moody, S. Hanson & R. Lippmann, eds. pp. 985-993. Morgan Kaufmann, San Mateo, CA, 1992.

Jordan, M. I.; Jacobs, R. A. Hierarchical Mixtures of experts and the EM algorithm. *Neural Computation*, vol. 6, pp. 181-214, MIT Press, 1994.

Jordan, M; Xu, L. Convergence results for the EM approach to mixtures of experts architectures. *Neural Networks,* vol. 8, no 9, pp. 1409-1431, 1995.

Kearns, M.; Mansour, Y.; Ng, A.; Ron, D. An experimental and theoretical comparison of model selection methods. *Machine Learning*, vol. 12, no 10, pp. 7-50, 1997.

Keerthi, S.; Shevade, S.; Bhattacharyya, C.; Murthy, K. Improvements to Platt'SMO algorithm for SVM classifier design. Tech. Report, Dept. of CSA, Banglore, India, 1999.

Keerthi, S. S.; Shevade, S. K.; Bhattachayya, C.; Murth, K. R. K. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, vol.13, pp. 637–649, March 2001.

Keerthi, S. S; Gilbert, E. G. Convergence of a generalized SMO algorithm for SVM classifier design. *Machine Learning*, vol. 46, no 3, pp. 351–360, March 2002.

Khotanzad, A; Chung, C. Hand written digit recognition using BKS combination of neural network classifiers. *Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation*, pp. 94-99, 1994.

Kim H-C.; Pang, S.; Je, H-M; Kim, D.; Bang, S-Y. Support vector machine ensemble with bagging, *Proceedings of International Workshop on Pattern Recognition with Support Vector Machines*, LNCS 2388, pp. 397-408, Canada, Springer, 2002.

Kohavi, R. **Wrappers for Performance Enhancement and Oblivious Decision Graphs**. Ph.D. Thesis, Stanford University, Computer Science Department, 1995.

Kohavi, R.; John, G. H. Wrappers for feature subset selection. *Artificial Intelligence*, vol. 97, no 1-2, pp. 273-324, 1997.

Krogh, A.; Vedelsby, J. Neural Network Ensembles, Cross Validation, and Active Learning. *Advances in Neural Information Processing Systems*, vol. 7, MIT press, Editors: Tesauro, G., Touretzky, D. S. and Leen, T. K. pp. 231-238, 1995.

Kuhn, H.; Tucker, A. Nonlinear programming. *Proceedings of 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, pp. 481–492. University of California Press, 1951.

Kullback, S.; Leibler, R. A. On Information and Sufficiency. *Annals of Mathematical Statistics*, vol. 22, pp. 79-86, 1951.

Kwok, J. T. Linear Dependency between ε and the Input Noise in ε–Support Vector Regression, in: G. Dorffner, H. Bishof, and K. Hornik (Eds), ICANN 2001, Lecture Notes on Computer Science (LNCS 2130), pp. 405-410, 2001.

Lang K.J. ; M. J. Witbrock. Learning to tell two spirals apart, *Proceedings of Connectionist Models Summer School*, pp. 52-59, Morgan Kaufmann, 1988.

Langley, P.; Simon, H. A. Applications of Machine Learning and Rule Induction, *Communications of the ACM*, vol. 38, no. 11, pp. 55-64, 1995.

Lapedes, A. S.; Faber, R. Nonlinear signal processing using neural networks prediction and system modeling. Technical Report LA-UR-87-2662. Los Alamos National Laboratory, Los Alamos, New Mexico 87545, 1987.

Laskov, P. An Improved Decomposition Algorithm for Regression Support Vector Machines. In Advances in *Neural Information Processing Systems*, vol. 12, S. A Solla, T. K. Leen and K. –R. Müller (eds), pp. 484-490, MIT Press, 2000.

LeBlanc, M.; Tibshiranin. Combining estimates in regression and classification. Paper available from ftp site: ustat.toronto.edu, 1993.

Lee, P. M. **Bayesian Statistics**. New York: Wiley, 1997.

Lee, S. I; Ahnn, J. H; Cho, S. B. Exploiting diversity of neural ensembles with speciated evolution. *of International Joint Conference on Neural Networks - IJCNN*, pp. 808-813, Washington, D. C., IEEE Press, 2001.

Lee D. S., Srihari S.N. Handprinted digit recognition: a comparison of algorithms. In: Srihari S. N. *et al.* (eds.) Proceedings of the Third International Workshop on Frontiers in Handwriting Recognition, pp. 153–164, 1993.

Lima C. A. M.; Coelho, A. L. V.; Von Zuben, F. J. Ensembles of Support Vector Machines for Regression Problems, *Proceedings of International Joint Conference on Neural Networks - IJCNN*, pp. 2381-2386, Hawaii, 2002.

Lima, C. A. M.; Coelho, A. L. V.; Villanueva, W. J. P.; Von Zuben, F. J. Gated Mixtures of Least Squares Support Vector Machine Experts Applied to Classification Problems. *Proceedings of the 5th International Conference on Recent Advances in Soft Computing* (RASC2004), Nottingham, United Kingdom, pp. 494-499, 2004.

Liu, Y.; Yao, X. Simultaneous Training of Negatively Correlated Neural Networks in an Ensemble. *IEEE Transactions on Systems Man and Cybernetics* – Part B, vol. 29, no 6, December, 1999.

Luenberguer, D. Linear and Nonlinear Programming. Addilson-Wesley, 1984

Luttrell, S. Self-Organized Modular Neural Networks for Encoding Data. Connection Science. *Special Issue on Combining Artificial Neural: Ensemble Approaches*, vol. 8, no 3 & 4, pp. 405-426, 1996.

MacKay, D. J. C. Comparison of approximate methods for handling hyper parameters. *Neural Computation*, In press, 1997.

Maclin, R.; Shavlik, J. W. Combining the predictions of multiple classifiers: using competitive learning to initialize neural networks. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Canada, Morgan Kaufmann, San Mateo, CA, pp. 524-530, 1995.

Mangasarian, O. L. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, vol. 13, pp. 444-452, 1964.

Mangazarian, O. L. **Nonlinear Programming**. McGraw-Hill, New York, NY, 1969.

Mao, J. A case study on bagging, boosting and basic ensembles of neural networks for OCR. *Proceedings of International Joint Conference on Neural Networks IJCNN*, vol. 3, Achorage, AK, *IEEE Computer Society Press*, Los Alamitos, CA, pp.1828-1833, 1998.

Mao, K. Z. Feature Subset Selection for Support Vector Machines Through Discriminative Function Pruning Analysis. *IEEE Transactions on Systems Man and Cybernetics – Part B*, vol. 34, no 1, february 2004.

Matterra, D.; Haykin, S. Support vector machines for dynamic reconstruction of a chaotic system. *In Advances in Kernel Methods (ed.* Schölkopf*, B., Burges, C. J. C. & Smola, A. J.)*, pp. 211–241. MIT Press, Cambridge, MA., 1999.

McCullagh, P.; Nelder, J. A. **Generalized Linear Models**. Chapman and Hall, London.

McLachlan, G. J.; Basford, K. E. **Mixture Models: Inference and Applications to Clustering**. Marcel Dekker, Inc., New York, 1988.

Merz, C. J.; Pazzani, M. J. Combining neural netowrk regression estimates with regularized linear weights. In M.C. Mozer, M. I. Jordan, T. Petsche (Eds.), Advances in *Neural Information Processing Systems 9*, Denver, CO, MIT Press, Cambridge, MA, pp. 564-570, 1997.

Moerland, P. Mixtures of experts estimate a posteriori probabilities. *IDIAP-RR 97-07, IDIAP*, Martigny, Switzerland, 1997.

Moerland, P. Some Methods for Training Mixtures of Experts. *IDIAP-Com 97-05*, 1997.

Moody, J. Fast learning in multi-resolution hierarchies. In D. S. Touretzky, editor, *Advances in Neural Information Processing System I*, chapter 1, pp. 29-39, Morgan Kaufmann, San Mateo, CA, 1989.

Moody, J.; Darken, C. Fast learning in networks of locally-tuned processing units. *Neural Computation*, vol. 1, pp. 281-294, 1989.

Mukherjee, S.; Tamayo, P.; Slonim, D.; Verri, A.; Golub, T.; Mesirov, J.; Poggio, T. Support vector machine classification of microarray data. AI Memo 1677, Massachusetts Institute of Technology, 1999.

Muller K.; Smola, A.; Ratsh, G.; Scholköpf, B.; Kohlmorgen, J.; Vapnik, V. Predicting time series with support vector machines, in: *Proceedings International Conference on Artificial Neural Networks ICANN*, LNCS 1327, pp. 999-1004, Springer, 1997.

Murphy, P.; Aha, D. UCI Repository of machine learning databases [Machine-readable data repository]. Irvine, CA: University of California, Department of Information and Computer Science. 1994.

Neal, R. M.; Hinton, G. E. A new view of the EM algorithm that justifies incremental and other variants. Unpublished manuscript from ftp://ftp.cs.utoronto.ca/pub/daford/em.ps.Z , 1993.

Nix, A. D.; Weigend, A. S. Learning local error bars for nonlinear regression. In G. Tesauro, D. S. Touretzky and T. K. Leen, editors, *Advances in Neural Information Processing Systems*, vol. 7, pp. 489-496, Cambridge MA, MIT Press, 1995.

Opitz, D. W.; Schavlik, J. W. Actively searching for an effective neural network ensemble. *Connection Science*, vol. 8, no 3 & 4, pp. 337-353, 1996.

Opitz, D. W.; Schavlik, J. W. Generating accurate and diverse members of a neural network ensemble. In D. S Touretzky, M. C. Mozer, M. E. Hasselmo (Eds.), Advances in *Neural Information Processing Systems 8*, Denver, CO, MIT Press, Cambridge, MA, pp. 535-541, 1996.

Opitz D. and R. Maclin. Popular ensemble methods: An empirical study, *Journal of Artificial Intelligence Research* 11, 169-198, 1999.

Oren, M.; Papageorgiou, C.; Sinha, P.;Osuna, E.; Poggio, T. Pedestrian detection using wavelet templates. *Proceeding Computer Vision and Pattern Recognition*, pp. 193–199, Puerto Rico, June 16–20 1997.

Osuna, E.; Freund, R.; Girosi, F. Training support vector machines: An application to face detection. In *Proceedings of the 1997 conference on Computer Vision and Pattern Recognition(CVPR'97)*, Puerto Rico, June, pp. 17-19 1997.

Osuna, E.; Girosi, F. Reducing the Run-time Complexity in Support Vector Machines, in B. Schölkopf, C. Burges and A. Smola (ed.), *Advances in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, pp. 185-208, 1999.

Papageorgiou C.; Evgeniou T.; Poggio T. A trainable pedestrian detection system. *Intelligent Vehicles*, pp. 241-246, 1998.

Parmanto, B.; Munro, P. W.; Doyle, H. R. Reducing variance of committee prediction with resampling technique. *Connection Science*. Special Issue on Combining Artificial Neural: Ensemble Approaches, vol. 8, no 3 & 4, pp. 405-426, 1996.

Partridge, D.; Griffith, N. Strategies for Improving Neural Net Generalization. *Neural Computing and Applications*, vol. 3, pp. 27-37, 1995.

Pavlov D.; Mao, J; Dom, B. Scaling-up support vector machines using boosting algorithm, *Proceeding 15th International Conference on Pattern Recognition - ICPR*, vol. 2, 219-222, Spain, 2000.

Perrone, M. P.; Cooper, L. N. When networks disagree: ensemble method for neural networks. In R. J. Mammone (Ed.), *Artificial Neural Networks for Speech and Vision*, Chapman & Hall, New York, pp. 126-142, 1993.

Peterson, G. E.; Barney, H. L. Control methods used in a study of vowels. *Journal of the Acoustical Society of America*, vol. 24, pp. 175-184, 1952.

Platt, J. Fast training of SVMs using sequential minimal optimization. In B. Schölkopf, C. Burges and A. Smola (ed.), *Advance in Kernel Methods: Support Vector Learning*, MIT press, Cambridge, MA, pp. 185-208, 1999.

Quinlan, J. R. Induction of decision trees. *Machine Learning*, vol. 1, pp. 81-106, 1986

Quinlan, J. R.; Rivest, R. L. Inferring decision trees using the Minimum Description Length Principle. *Information and Computation*, vol. 80, pp. 227-248, 1989.

Ramamurt, V.; Ghosh, J. Regularization and Error Bars for the Mixture of Experts Network. In the *Proceedings of the IEEE International Conference on Neural Networks* 1997.

Ramamurt, V; Ghosh, J. Structurally Adaptive Localized Mixtures of Experts for Non-Stacionary Environments. *IEEE Transactions Neural Networks*; vol. 10, no 1, pp. 152-160, 1999.

Rasmussen, C. E. Evaluation of Gaussian Processes and other methods for non-linear regression, PhD thesis, University of Toronto, 1996.

Rätsch, G.; Onoda, T.; Müller, K. R. Regularizing Adaboost, *Advances in Neural Information Processing Systems 11*, Kearns, M. S., Solla, S. A., and Cohn, D. A. Eds., MIT Press, 564, 1999.

Rätsch, G.; Schölkopf, B.; Smola, A.; Müller, K. –R.; Onoda, T.; Mika, A. v-Arc: ensemble learning in the presence of noise. *Advances in Neural Information Processing Systems* 12, Solla, S. A., Leen T. K., and Müller, K. R. (Ed.), MIT Press, 561, 2000.

Raviv, Y.; Intrator, N. Variance Reduction via Noise and Bias in constraints. *In Combining Artificial Neural Nets: Ensemble and Modular Multi-Net Systems,* ed. Amanda Sharkey*,* Springer-Verlag London Ltd, 1999.

Reily, R. L; Scofield, C. L.; Elbaum, C.; Cooper, L. N. Learning system architectures composed of multiple learning modules. *Proceeding First International Conference on Neural Networks – ICNN*, vol. 2, IEEE, 1987.

Reily, R. L.; Scofield, C. L.; Cooper, L. N.; Elbaum, C. Gensep: A multiple neural network learning system with modifiable network topology. *Abstracts of the First Annual International Neural Network Society Meeting*. INNS, 1988.

Rida, A.; A. Labbi; C. Pellegini. Local Experts Combination through Density Decomposition. *Proceedings of International Workshop on Artificial Intelligence and Statistics* (Uncertainty'99), Morgan Kaufmann, 1999.

Rogova, G. Combining the results of several neural network classifiers. *Neural Networks*, vol. 7, no 5, pp. 777-791, 1994.

Rosseeuw, P. J.; Leroy, A. Robust regression and outliers detection. New York: Wiley, 1987.

Rosseeuw, P. J.; Van Zomeren, B. C. Unmasking multivariate outliers and leverage points. *Journal of the American Statistical Association*, vol. 85, pp. 633-639, 1990.

Saito, K.; Nakano, R. A constructive learning algorithm for na HME. In *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1268-1273, 1996.

Sanger, T. D. A tree-structured adaptive network for function approximate in high-dimensional spaces. *IEEE Transactions on Neural Networks*, vol. 2, no 2, pp. 285-293, march 1991.

Saunders, C.; Gammerman, A.; Vovk, V. Ridge Regression Learning Algorithm in Dual Variables. Proc. of the 15th *Int. Conf. On Machine Learning* (ICML'98), Morgan Kaufmann, pp. 515-521, 1998.

Schapire, R. E. The strength of weak learnability. *Machine Learning*, vol. 5, pp. 197-227, 1990.

Schapire, R. E.; Freund, Y.; Bartlett, P.; Lee, W. S. Boosting the margin: A new explanation for the effectiveness of voting methods. The *Annals of Statistics*, 1651, 1998.

Schölkopf, B. **Support Vector Learning**. Munich, Germany: Oldenburg-Verlag, 1997.

Schölkopf, B.; Platt, J. C.; Shawe-Taylor, J.; Smola, A. J.; Wiliamson, R. C. Estimating the support of a high-dimensional distribution. *Microsoft Research Corporation Report* MSR-TR-99-87, 1999.

Schölkopf, B.; Shawe-Taylor, J.; Smola, A; Williamson, R. Kernel-dependent support vector error bounds, *Ninth International Conference on Artificial Neural Networks*, IEEE Conference Publications no 470, pp. 304-309, 1999.

Schölkopf, B.; Herbrich, R.; Smola, A. J. A generalized representer theorem. *Proceedings of the Annual Conference on Computational Learning Theory*. 2001.

Schölkopf, B.; Smola, A. **Learning with kernels**, MIT Press, Cambridge, MA, 2002.

Schwaighofer A.; Tresp, V. The Bayesian committee support vector machine. *Proceedings International Conference on Artificial Neural Networks - ICANN*, LNCS 2130, 411-417, Austria, Springer, 2001.

Schapire, R. E. The strength of weak learn ability. *Machine Learning*, vol. 5, no 2, pp. 197-227, 1990.

Sharkey, A. J. C.; Sharkey, N. E.; Chandroth, G. O. Neural nets and diversity. Neural Computing and Application, vol 4, pp. 218-227, 1996.

Sharkey, A. J. C.; Sharkey, N. E. Combining diverse neural nets. The Knowledge Engineering Review, vol. 12, no 3, pp. 231-247, 1997.

Sharkey A. (Ed.). Combining artificial neural nets*:* Ensemble and modular multi-net systems, Springer-Verlag, London, 1999.

Shimshoni, Y., Intrator, N. Classification of seismic signals by integrating ensembles of neural networks. *IEEE Transactions Signal Processing* vol. 46, no 5, pp. 1194-1201, 1998.

Slonim D.; Tamayo P.; Mesirov J.; Golub T.; Lander E. Class prediction and discovery using gene expression data. *Proceeding of the 4th Annual International Conference on Computational Molecular Biology - RECOMB*, Universal Academy Press, pp. 263-272, Tokyo, Japan. 2000.

Smits G. F.; Jordaan E. M. Improved SVM regression using Mixture of Kernels, *Proceeding of International Joint Conference on Neural Networks - IJCNN*, 2785-2790, Hawaii, 2002.

Smola, A; Schölkopf, B. A tutorial on support vector regression. *NeuroColt2* TR 1998-03, 1998.

Smola, A. **Learning with kernels**. Phd Thesis, published by: GMD, Birlinghoven, 1999.

Sollich, P.; Krogh, A. Learning with ensembles: how over-fitting can be usefull. In D. S. Touretzky, M. C. Mozer, M. E. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 8*, Denver, CO, MIT Press, Cambridge, MA, pp. 190-196, 1996

Suykens, J. A. K.; Vanderwalle, J. Least squares supporte machine classifiers. *Neural Processing Letters*, vol. 9, no 3, pp. 293-300, June, 1999.

Suykens, J. A. K.; Vandewalle, J. Recurrent least squares support vector machines. *IEEE Transactions on Circuits and Systems-I*, vol. 47, pp. 1109–1114, 2000.

Suykens, J. A. K.; Van Gestel, T.; De Brabanter, J.; De Moor, B.; Vanthienen, J. Least Squares Support Vector Machines. *World Scientific*, 2002.

Taniguchi, M.; Tresp, V. Averaging regularized estimators. *Neural Computation*, vol. 9, 1163, 1997.

Hastie, T.; Tibshirani, R.; Friedman, J. H. **The Elements of Statistical Learning**, 2001.

Tikhonov, A. N.; Arsenim, V.Y. **Solutions of Ill-posed Problems**. W. H. Winston, Washington, D.C., 1997.

Tin-Yau Kwok J. Support vector mixture for classification and regression problems, *Proceeding 13th International Conference on Pattern Recognition - ICPR*, 255-258, Brisbane, 1998.

Titterington, D. M.; Smith, A. F. M.; Makov, U. E. **Statistical Analysis of Finite Mixture Distributions**, John Wiley, New York, 1985.

Tresp V. Committee machines. In Y. H. Hu and J.-N. Hwang, eds., Handbook for Neural Signal Processing, CRC Press, 2001.

Tumer, K.; Ghosh, J. Order statistics combiners for neural classifiers. *Proceedings of the World Congress on Neural Networks*, pp. I 31:34. INNS, Press, Washington DC, 1995.

Tumer, K.; Ghosh, J. Error correlation and error reduction in ensemble classifiers. Connection Science. *Special Issue on Combining Artificial Neural: Ensemble Approaches*, vol. 8, no 3 & 4, pp. 385-404, 1996.

Valentini G.; Dietterich, T. Bias-variance analysis and ensembles of SVM, in: Proc. of *Third International Workshop on Multiple Classifier Systems* (MCS), LNCS 2364, 222-231, Italy, Springer, 2002.

Vapnik, V. N; Lerner, A. Pattern recognition using generalized portrait method. *Automation and Remote Control*, vol. 24, 1963.

Vapnik, V. N. **Estimation of Dependences Based on Empirical Data**. Springer-Verlag, Berlim, 1982.

Valiant, L. A theory of the learnable. *Communications of the ACM*, 27, 1984.

Vapnik V.N. **The nature of statistical learning theory**, Springer-Verlag, 1995.

Vapnik, V.; Golowich, S.; Smola, A. Support Vector Method for Function Aproximation, Regression Estimation, and Signal Processing. In M. Mozer, M. Jordan, and T. Petsche (Eds): *Neural Information Processing Systems*, vol. 9, MIT Press, Cambridge, MA, 1997.

Vapnik V. N. **Statistical Learning Theory**, John Willey & Sons, 1998.

Vapnik, V. N. An overview of statistical learning theory. *IEEE Transactions Neural Networks*, vol. 10, pp. 988-999, Sept. 1999.

Waterhouse, S. R.; Robinson, A. J. Classification using hierarchical mixtures of experts. *In Proceedings 1994 IEEE Workshop on Neural Networks for Signal Processing*, pp. 177-186, Long Beach CA, IEEE Press, 1994.

Waterhouse, S. R.; Robinson, A. J. Constructive algorithms for hierarchical mixtures of experts. *Advances in Neural Information Processing Systems 8*, Eds. Touretzky, D. S; Mozer, M. C. Hasselmo, M. E, MIT Press, 55 Hayward St., Cambridge, MA, 02142-1399, pp. 584-590, 1996.

Waterhouse, S. R.; MacKay, D. J. C., Robinson, A. J. Bayesian methods for mixture of experts. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, Advances in *Neural Information Processing Systems*, vol. 8, pp. 351-357, Cambridge MA, MIT Press, 1996.

Waterhouse, S. R. Classification and Regression using Mixtures of Experts. Doctor' thesis Jesus College, Cambridge, Departament of Engineering, University of Cambridge, 1998.

Weigend, A. S.; Huberman, B. A.; Rumelhart, D. Predicting the future: A connectionist approach. *International Journal of Neural Systems*, vol. 1, pp. 193-209, 1991.

Weigend A.; Gershenfeld N. (Eds.), Time Series Prediction: Forecasting the Future and Understanding the Past, Reading, MA: Addison-Wesley, 1994.

Weigend, A. S.; Mangeas, M.; Sristava, A. N. Nonlinear gated experts for time series: Discovering regimes and avoiding over fitting. *International Journal of Neural Systems*, vol. 6, pp. 373-399, 1995.

Wernecke, K-D. A coupling procedure for the discrimination of mixed data. *Biometrics*, vol. 48, pp. 497-506, 1992.

Weston, J.; Mukherjee, S.; Chapelle, O.; Pontil, M.; Poggio T.; Vapnik, V. Feature Selection for SVMs. *Advances in Neural Information Processing Systems* vol. 13, 2000.

Weston, J.; Murkherjee, S.; Chapelle, O.; Pontil, M.; Poggio, T.; Vapnik, V. N. Feature selection for SVMs. *Advances in Neural Information Processing Systems*, S. A. Solla, T. K. Leen, and K. –R. Muller, Eds. Cambridge, MA: MIT Press, vol. 13, 2001.

Widrow B. and Stearns, S. **Adaptive Signal Processing**, Prentice-Hall, USA, 1985.

Williams, P. M. Using neural networks to model conditional multivariate densities. *Neural Computation*, vol. 8, no 4, pp. 843-854, 1996.

Wolpert, D. H. Stacked generalization. *Neural Networks*, vol 5, no 3, pp. 241-259, 1992.

Xu, L.; Krzyzakk, A.; Suen, Y. C. Several methods for combining multiple classifiers and their applications in handwritten character recognition. *IEEE Transactions on System*, Man and Cybernetics, vol. 22, no 3, pp. 418-435. 1992.

Xu, L; Jordan, M. I; Hinton, G. E. An alternative model for mixture of experts. In G. Tesauro, D. S. Touretsky, and T. K. Leen, editors. *Advances in NIPS*, vol. 7, pp. 633-640, Cambridge MA, MIT Press, 1995.

Yang, Y.; Slattery, S.; Ghani, R. A Study of Approaches to Hypertext Categorization, *Journal of Intelligent Information Systems,* vol. 18, no 2, March 2002.

Yao, X.; Liu, Y. Making use of population information in evolutionary artificial neural networks. *IEEE Transactions on Systems*, Man and Cybernetics-Part B: Cybernetics, vol. 28, no 3, pp. 417-424, 1998.

Zemel, R. S.; Pitassi, T. A gradient-based boosting algorithm for regression problems. *Advances in Neural Information Processing Systems* 13, Leen, T. K., Dietterich, T. G., and Tresp V., Eds., MIT press, 2001.

Zeng, X.; Martinez, T. R. Using a Neural Network to Approximate an Ensemble of Classifiers. *Neural Processing Letter*, vol. 12, pp. 225-237, 2000.

Zhilkin, P. A.; Somorjai, R. L. Application of several methods of classification fusion to magnetic resonance spectra. Connection Science. *Special issue on Combining Artificial Neural Nets: Ensemble Approaches*, vol. 8, no 3 & 4, pp. 427-442, 1996.

Zhou, Z. H.; Jiang, Y.; Yang, Y. B.; Chen, S. F. Lung cancer cell identification based on neural network ensembles. *Artificial Intelligence in Medicine*, vol. 24, no 1, pp. 25-36, 2000.

Zhou, Z. H.; Wu, J.; Tang, W. Ensembling Neural Networks: Many Could Be Better Than All. *Artificial Intelligence*, vol. 137, no 1-2, pp. 239-263, 2002.