



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA DE TELECOMUNICAÇÕES
CAMPUS DE FORTALEZA

Shara Shami Araújo Alves

**SOLUÇÃO DO PROBLEMA DE OTIMIZAÇÃO QUADRÁTICA
EM MÁQUINAS DE VETORES SUPORTE PARA REGRESSÃO
VIA ALGORITMOS GENÉTICOS**

Fortaleza - Ceará
Fevereiro 2017

Shara Shami Araújo Alves

**SOLUÇÃO DO PROBLEMA DE OTIMIZAÇÃO QUADRÁTICA
EM MÁQUINAS DE VETORES SUPORTE PARA REGRESSÃO
VIA ALGORITMOS GENÉTICOS**

Dissertação submetida à Coordenação do Programa de Pós-Graduação em Engenharia de Telecomunicações do Instituto Federal do Ceará, como requisito parcial para obtenção do grau de Mestre em Engenharia de Telecomunicações.

Área de concentração: Sistemas de Telecomunicações.

Orientador: Prof. Dr. Ajalmar R. da Rocha Neto

Fortaleza
2017

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

ALVES, SHARA SHAMI ARAUJO ALVES.

SOLUÇÃO DO PROBLEMA DE OTIMIZAÇÃO QUADRÁTICA EM MÁQUINAS DE VETORES
SUPORTE PARA REGRESSÃO VIA ALGORITMOS GENÉTICOS / SHARA SHAMI ARAUJO ALVES
ALVES. - 2017.

82 f. : il. color.

Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Engenharia de Telecomunicações,
Campus Fortaleza, 2017.

Orientação: Prof. Dr. Ajalmar R. da Rocha Neto.

1. support vector machines. 2. regression. 3. evolutionary machine learning. 4. genetic algorithms. I.
Titulo.

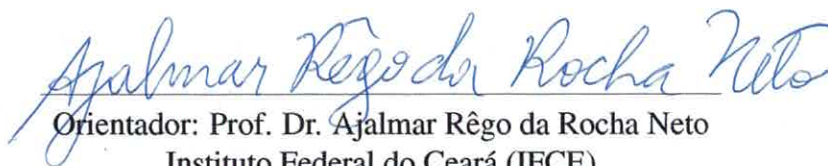
CDD 621.382

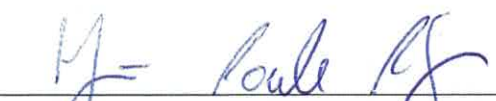


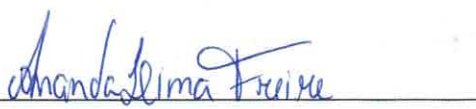
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELECOMUNICAÇÕES

Shara Shami Araújo Alves

Esta dissertação foi julgada adequada para a obtenção do título de Mestre em Engenharia de Telecomunicações, sendo aprovada pela Coordenação do Programa de Pós-Graduação em Engenharia de Telecomunicações do Instituto Federal de Educação, Ciência e Tecnologia do Ceará e pela banca examinadora:


Orientador: Prof. Dr. Ajalmar Rêgo da Rocha Neto
Instituto Federal do Ceará (IFCE)


Prof. Dr. João Paulo Pordeus Gomes
Universidade Federal do Ceará (UFC)


Prof. Dr. Ananda Lima Freire
Instituto Federal do Ceará (IFCE)


Prof. Dr. Regis Cristiano Pinheiro Marques
Instituto Federal do Ceará (IFCE)

Fortaleza, 21 de Fevereiro de 2017

Este trabalho é dedicado à minha mãe, Francisca Eliane Diniz Araújo, pelo exemplo de vida, dedicação aos filhos e por sempre me apoiar nas minhas escolhas. Dedico também aos meus irmãos, Joshua Ben Rabi Araújo Alves e Jeshua Ben-Ami Araújo Alves, a meu querido companheiro Pedro Vinnicius Vieira Alves Cabral por toda paciência e persistência. Por fim, dedico aos meus amigos do Laboratório de Processamento de Sinais (PDS), Saulo Anderson Freitas de Oliveira, Alan Magalhães Braga, Madson Luiz Dantas Dias, Elizângela de Souza Rebouças, David Clifte da Silva Vieira e Lucas Silva de Sousa, por me apoiarem e me ajudarem durante a construção deste trabalho.

Agradecimentos

Primeiramente a Deus, pois sem sua permissão nada seria possível.

A minha mãe por me motivar a estudar desde cedo e sempre ser uma fonte eterna de incentivo.

Ao meu querido e amado companheiro, Pedro Cabral, por sempre estar ao meu lado, me apoiando, me incentivando e me ajudando. Os dias e noites de estudo e trabalho com sua presença e apoio fizeram toda a diferença nesta caminhada.

Ao meu orientador, Prof. Dr. Ajalmar Rêgo da Rocha Neto, pela orientação, conhecimentos e entre muitas outras qualidades, pela perseverança em incentivar-me nos momentos cruciais para execução deste trabalho.

Aos meus amigos pesquisadores Alan Magalhães Braga, Saulo Anderson Freitas de Oliveira, Madson Luiz Dantas Dias, Elizângela de Souza Rebouças, David Clifte da Silva Vieira e Lucas Silva de Sousa por tornarem a convivência no laboratório PDS uma experiência única e enriquecedora, pelos momentos de brincadeira, conversas soltas e sempre me incentivarem nos momentos mais difíceis da minha vida acadêmica.

Ao Laboratório de Processamento de Sinais (PDS).

A CAPES pelo apoio financeiro;

Aos demais professores do IFCE, as secretárias da pós-graduação e a todos os funcionários do departamento pela amizade e colaboração durante o decorrer do mestrado.

A todos que de alguma forma contribuíram para a realização deste trabalho.

”A amizade é um amor que nunca morre.”

Mário Quintana

Resumo

As Máquinas de Vetores Suporte (SVM), amplamente difundidas em reconhecimento de padrões, possuem uma formulação aplicada às tarefas de regressão, chamada de Regressores de Vetores Suporte (SVR). O modelo é obtido pela resolução de um problema de otimização quadrática que envolve encontrar os valores dos multiplicadores de Lagrange e o viés. Os métodos amplamente utilizados, considerados estado da arte, para resolução de tal problema de otimização quadrática são, a saber: *Sequential Minimal Optimization* (SMO), *Iterative Single Data Algorithm* (ISDA) e métodos clássicos de Programação Quadrática (QP). Estratégias evolucionárias, como têmpera simulada, programação genética, algoritmos genéticos dentre outros, quando combinadas com aprendizado de máquina compreendem a área chamada Aprendizado de Máquina Evolucionário. Os trabalhos nesta área focam, prioritariamente, em três aspectos: otimização de parâmetros, seleção de atributos ou ainda encontrar um conjunto reduzido de vetores suporte em SVMs. Esta dissertação propõe resolver a formulação dual do problema de otimização quadrática inerente à SVR através de Algoritmos Genéticos, que são metaheurísticas que exploram o espaço de busca de forma não exaustiva e são comumente aplicadas em problemas de otimização. Neste trabalho foi realizado um estudo comparativo dos métodos de treinamento de SVR e apresentada uma nova proposta de treinamento, denominada Regressor de Vetor Suporte Evolucionário. A proposta foi validada em conjuntos de dados reais e seus resultados comparados com os métodos estado da arte mencionados anteriormente (SMO, ISDA, QP). A análise dos resultados revelaram que a proposta se mostra bastante promissora, ao passo que reduz o erro de generalização e aumenta a esparsidade na maioria dos experimentos realizados.

Palavras-Chave: *regressor de vetor suporte, aprendizado de máquina evolucionário, algoritmos genéticos.*

Abstract

Support Vector Machines which are widespread in pattern recognition have also a formulation for regression tasks, the so-called Support Vector Regression (SVR). The model is obtained by solving a quadratic optimization problem that involves finding the values of the Lagrange multipliers as well as the bias. The widely used methods, considered state-of-the-art, to solve such quadratic optimization problem, are: Sequential Minimal Optimization (SMO), Iterative Single Data Algorithm (ISDA) and classical methods of Quadratic Programming (QP). Evolutionary strategies such as simulated annealing, genetic programming and genetic algorithms when combined with machine learning comprehend the Evolutionary Machine Learning area. Those works focus, in priority, on three aspects: parameters optimization, attributes selection and to find a reduced set of support vectors. This work aims to solve the problem of quadratic optimization in its dual formulation through Genetic Algorithms, which are metaheuristics that can be applied in optimization problems and that explore the search space in a non-exhaustive way. In this work, we made a comparative study of the aforementioned well-known SVR training methods, SMO, QP and ISDA; and we also presented a new training method, named Evolutionary Support Vector Regression. The proposal was validated with real-world datasets and its results were compared with the state-of-the-art methods mentioned previously (SMO, ISDA, QP). The results revealed that the proposal is very promising, while reducing the generalization error and increasing the sparsity in most experiments.

Keywords: *support vector machines, regression, evolutionary machine learning, genetic algorithms.*

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Símbolos

Lista de Abreviações

1	Introdução	16
1.1	Motivação	17
1.2	Objetivo geral	18
1.3	Objetivos específicos	18
1.4	Produção científica	19
1.5	Organização da Dissertação	19
2	Algoritmos Genéticos	21
2.1	Introdução	21
2.2	Visão geral	22
2.3	Representação genética	24
2.4	População	25
2.5	Operadores Genéticos	25
2.5.1	Operadores de Seleção	26
2.5.2	Elitismo	26
2.5.3	Operadores de Reprodução	27

2.5.4	Operadores de Mutação	29
2.6	Função de aptidão (<i>fitness</i>)	30
2.7	Crítérios de convergência	30
2.8	Conclusão	31
3	Máquinas de Vetores Suporte para regressão	32
3.1	Introdução	32
3.2	Classificadores de Vetores Suporte	33
3.3	Regressores de Vetores Suporte	37
3.4	Métodos de treinamento	43
3.4.1	Programação Quadrática	44
3.4.2	<i>Sequential Minimal Optimization</i>	45
3.4.3	<i>Iterative single data algorithm</i>	47
3.4.4	ϵ -evolutionary support vector regression	51
3.5	Crítérios de convergência	51
3.6	Conclusão	52
4	Algoritmos Genéticos no treinamento de SVR	53
4.1	Visão geral	53
4.2	População e representação genética	54
4.3	Algoritmo de ajuste	56
4.4	Função de aptidão	58
4.5	Operadores genéticos	58
4.6	Elitismo	59
4.7	Cômputo do termo viés	59
4.8	Conclusão	59
5	Simulações Computacionais	60

5.1	Metodologia	60
5.2	Resultados e discussões	63
5.2.1	Conclusão	74
6	Conclusões e Trabalhos Futuros	75
6.1	Contribuições deste Trabalho	76
6.2	Trabalhos Futuros	76
	Referências	77

Lista de Figuras

1	Execução de um Algoritmo Genético	23
2	Exemplos de diferentes codificações de indivíduos em Algoritmos Genéticos. .	24
3	Funcionamento do <i>crossover</i> de 1 ponto	27
4	Funcionamento do <i>crossover</i> multi-ponto,	27
5	Funcionamento do <i>crossover</i> uniforme	28
6	Funcionamento do <i>crossover</i> OX	28
7	Exemplo de escolha do hiperplano com SVM linear.	33
8	Arquitetura de um regressor de vetor suporte.	38
9	Variáveis de folga em SVR e função de perda ϵ -insensitive.	39
10	Aproximação da função sinc x com precisão $\epsilon \in 0,1;0,2;0,5$. Em (a), (c) e (e): a linha sólida superior e a linha tracejada inferior indica o tamanho do tubo ϵ , a linha pontilhada é $f(\mathbf{x})$. Em (b), (d) e (f): a relação entre a aproximação (linha sólida) e a esparsidade. Os dados são os pontos menores e os vetores suportes os pontos maiores.	40
11	Fluxo do ESVR.	54
12	Representação genética, considerando o parâmetro $C = 4$	55
13	Aplicação do algoritmo de ajuste. Considere o parâmetro $C = 4$	56
14	Relação MSE e SV nos conjuntos SER, AIL e ELE com <i>kernel</i> linear.	65
15	Relação MSE e SV para os conjuntos de dados SER, AIL e ELE com <i>kernel</i> gaussiano.	68
16	Relação MSE e SV para os conjuntos de dados AUT, CPU, SER, BCR, BOS e MOT com <i>kernel</i> linear e gaussiano.	69

17	Relação MSE e SV nos conjuntos de dados AIL e ELE com <i>kernel</i> linear e gaussiano.	70
18	Conjunto de dados MOT com os métodos SMO, QP, ISDA e ESVR com <i>kernel</i> gaussiano. Círculos vazados são os padrões cujo $\alpha_i > 0$ e quadrados vazados se $\alpha_i^* > 0$	71
19	Relação do MSE normalizado e relação de vetores suporte conjunto/método com <i>kernel</i> linear e gaussiano.	73

Lista de Tabelas

1	Detalhes dos conjuntos de dados das simulações	60
2	Parâmetros dos métodos	63
3	Resultados da média das 30 realizações com <i>kernel</i> linear.	64
4	Resultados da média das 30 realizações com <i>kernel</i> gaussiano.	66

Lista de Símbolos

$\mathbf{x}_i$	Vetor de atributos de entrada
y	Variável correspondente
$(2\mathbf{x}_i, y)$	Padrão
l	Tamanho do conjunto de treinamento
$\mathbb{R}$	Conjunto dos números Reais
P	Dimensão do padrão
$f(.)$	Função
$\mathbf{w}$	Vetor de pesos
b	Viés
$ \cdot $	Norma euclidiana
$L(.)$	Função lagrangiana
ε	desvio
$\boldsymbol{\alpha}, \boldsymbol{\beta}$	Vetores de multiplicadores de Lagrange
ξ	Variável de folga
C	Parâmetro de regularização de SVMs
$K(.,.)$	Função de <i>kernel</i>
σ	Desvio padrão

Lista de Abreviações

AG	Algoritmos Genéticos
ESVR	Regressor de Vetor Suporte Evolucionário
ISDA	<i>Iterative Single Data Algorithm</i>
KKT	<i>Karush-Kuhn-Tucker</i>
QP	Programação Quadrática
RBF	<i>Radial-Basis Function</i>
RNA	<i>Artificial Neural Network</i>
SVC	<i>Support Vector Classifier</i>
SVM	<i>Support Vector Machines</i>
SMO	<i>Sequential Minimal Optimization</i>
SVR	<i>Support Vector Regressor</i>

INTRODUÇÃO

Regressão é uma técnica que permite explorar e inferir a relação de uma variável dependente (variável de resposta) com variáveis independentes, ou seja, a regressão é responsável por encontrar um modelo que melhor represente a relação entre duas ou mais variáveis. Dentre as diversas ferramentas existentes, a aproximação de funções pode ser obtida a partir de uma rede neural Perceptron multicamadas, rede de função de base radial (*Radial-Basis Function*, RBF), modelos *fuzzy*, função de aproximação polinomial, etc. As Máquinas de Vetores Suporte (*Support Vector Machines*, SVM), nas quais o modelo é obtido pela resolução de um problema de otimização quadrática, fazem parte das técnicas em aprendizado de máquina que podem ser aplicadas para tarefas de regressão.

Máquinas de Vetores Suporte foram primeiramente introduzidas por Vapnik (BOSER; GUYON; VAPNIK, 1992). SVMs foram desenvolvidas seguindo a ordem reversa em relação as Redes Neurais Artificiais (*Artificial Neural Network*, RNA) (HOPFIELD, 1982), pois partiram da teoria para a implementação e experimentos, enquanto que RNAs seguiram um caminho mais heurístico, da aplicação e experimentação para a teoria. Apesar de SVMs possuírem uma teoria bem definida, só chamaram atenção de fato quando foram obtidos resultados excelentes na aplicação de SVMs em reconhecimento numérico, visão computacional e categorização de textos (KECMAN; HUANG; VOGT, 2005).

Além disso, SVMs ganharam popularidade devido a sua habilidade em minimizar ambos os riscos empírico e o estrutural. Tal característica tem se mostrado superior à minimização do risco empírico utilizado em redes neurais artificiais tradicionais. SVMs constituem técnicas úteis tanto para tarefas de classificação quanto de regressão, quando aplicadas à classificação de padrões são conhecidas como classificadores de vetores suporte (*Support Vector Classifiers*, SVCs) e regressores de vetores suporte (*Support Vector Regressors*, SVRs) (VAPNIK, 1995) quando aplicadas à regressão.

O processo de treinamento de um SVR requer a solução de um problema de otimização quadrática, o qual pode ser descrito em sua formulação primal e dual. A formulação dual é apresentada em termos de um conjunto de multiplicadores de Lagrange e o viés (*bias*) (SMOLA;

SCHÖLKOPF, 2004). Há duas abordagens bastante utilizadas para a resolução do problema de otimização quadrática, a primeira é baseada em métodos clássicos matemáticos e a segunda em decomposição do conjunto de dados onde a solução é obtida iterativamente ao invés de diretamente como na maioria dos métodos clássicos.

Os métodos de treinamento considerados estado da arte são os algoritmos de decomposição de conjunto de dados: *Sequential Minimal Optimization* (SMO) (PLATT, 1999) e *Iterative Single Data Algorithm* (ISDA) (KECMAN; HUANG; VOGT, 2005), e o método clássico de Programação Quadrática (QP) (GILL; MURRAY; WRIGHT, 1981). Porém, os métodos clássicos matemáticos, como QP, requerem a manipulação de grandes matrizes e podem gerar erros de precisão numérica, visto que QP resolve o problema através de otimização numérica.

1.1 Motivação

A área de aprendizado de máquina evolucionário tem ganhado popularidade na comunidade de aprendizado de máquina. Esta área engloba qualquer combinação de técnicas de aprendizado de máquina (redes neurais, classificadores, regressores, dentre outros) com computação evolucionária, tais como têmpera simulada (LAARHOVEN; AARTS, 1987), programação genética (BANZHAF *et al.*, 1998), otimização por enxame de partículas (KENNEDY, 2011), e algoritmos genéticos (HOLLAND, 1992). Neste contexto, metaheurísticas são alternativas para métodos com base em otimização numérica uma vez que elas são capazes de solucionar problemas de otimização sem suposição de linearidade, diferenciabilidade, continuidade ou convexidade da função objetivo (SILVA; SILVA; ROCHA NETO, 2015).

Algoritmo Genético (AG) (HOLLAND, 1992) é um método de busca metaheurístico que pode ser utilizado para gerar soluções úteis para problemas de otimização ou em problemas em que o espaço a ser explorado é demasiado grande e seria impraticável explorá-lo por uma abordagem exaustiva. AGs exploram o espaço de soluções de maneira direcionada utilizando a informação pertinente ao problema alinhada com algumas componentes aleatórias. Diferenciam-se de modelos estritamente aleatórios pois, as componentes aleatórias não são as principais condutoras da exploração.

AGs têm sido amplamente aplicados para lidar com otimização de parâmetros (CHEN; WANG, 2007; GASCÓN-MORENO *et al.*, 2011), seleção de características (MONTEIRO; KOSUGI, 2007; CERVANTE *et al.*, 2012; KHARRAT; HALIMA; AYED, 2015), treinamento de RNAs (ZHANG *et al.*, 2016; IJJINA; CHALAVADI, 2016; JIA; ZHAO; DING, 2016) e para obter um conjunto reduzido de vetores de suporte em SVMs (WANG; XU, 2004; ROCHA

NETO; BARRETO, 2013; SILVA; SILVA; ROCHA NETO, 2015). No entanto, trabalhos que realizem o treinamento de SVMs através de AGs são poucos na literatura. Os poucos trabalhos que focam em resolver o problema de otimização quadrática inerente à SVCs em sua formulação primal foram encontrados em Dumitrescu *et al.* (2006) e em Stoean *et al.* (2009), e em sua formulação dual em Dias e Rocha Neto (2016) e em Mierswa (2006). Somente em Dias e Rocha Neto (2016), todas as restrições inerentes à SVCs foram atendidas.

É importante ressaltar que são poucas as abordagens na literatura que resolvam o problema de otimização quadrática inerente à SVRs. Uma abordagem que utiliza AGs foi encontrada em Stoean *et al.* (2006), porém é restrita a formulação primal do problema de otimização quadrática. Este trabalho supracitado realizou experimentos modestos e utilizou uma metodologia fraca, visto que seus experimentos foram realizados sobre apenas uma base de dados e as comparações realizadas não compreendem o estado da arte. Além disso, por se tratar do problema primal, a abordagem é limitada ao kernel linear. Uma outra vertente pouco difundida na literatura atual, objeto de estudo e contribuição desta dissertação, consiste na utilização de Algoritmos Genéticos com foco no treinamento de SVMs para regressão na sua formulação dual, de tal maneira que as restrições do problema sejam satisfeitas.

Desta forma, esta dissertação propõe uma abordagem para a resolução do problema de otimização quadrática inerente à SVRs através de AGs. Tal abordagem é avaliada sobre conjuntos de dados reais e seus resultados comparados à métodos comumente utilizados no treinamento de SVRs. Através da análise dos resultado é possível constatar que a proposta é promissora.

1.2 Objetivo geral

O objetivo deste trabalho é realizar um estudo comparativo entre os métodos de treinamento de SVRs. Além disso, propor e avaliar o uso de Algoritmos Genéticos no treinamento de SVRs para resolver o problema de otimização quadrática na sua formulação dual.

1.3 Objetivos específicos

Essa dissertação tem os seguintes objetivos específicos:

- Realizar um estudo comparativo de métodos que visam a solução do problema de otimização quadrática na sua formulação dual em SVR;

- Realizar a modelagem da formulação dual do problema de otimização quadrática inerente à SVRs para o arcabouço de Algoritmos Genéticos, no tocante a representação genética, operadores de reprodução e mutação, assim como a modelagem da função de avaliação;
- Analisar os resultados obtidos a partir da proposta em comparação com outros métodos pertencentes ao estado da arte, tais como *Sequential Minimal Optimization*, Programação Quadrática e o *Iterative single data algorithm*. Avaliar o desempenho por meio das métricas de erro quadrático médio, número de vetores suporte, assim como o custo computacional de treinamento e teste.

1.4 Produção científica

Os seguintes artigos foram aceitos em conferências e publicados como capítulo de livro no transcorrer deste trabalho, a saber:

- **Shara S. A. Alves**, Saulo A. F. Oliveira e Ajalmar R. Rocha Neto. “A Novel Educational Timetabling Solution through Recursive Genetic Algorithms”. *2nd Latin American Congress on Computational Intelligence, LA-CCI. 2015. (Conferência);*
- **Shara S. A. Alves**, Saulo A. F. Oliveira e Ajalmar R. Rocha Neto. ‘A Recursive Genetic Algorithm-based approach for Educational Timetabling Problems”. *Designing with Computational Intelligence, Springer Book Series Studies in Computational Intelligence. 2017. (Capítulo de livro);*

Além disso foram submetidos o seguinte artigo:

- Saulo A. F. Oliveira, **Shara S. A. Alves**, João P. P. Gomes e Ajalmar R. Rocha Neto. “A Fixation Prediction-based Approach for Image Retargeting Quality Assessment”. *Computer Vision and Image Understanding. 2017. (Revista).*

1.5 Organização da Dissertação

Este trabalho é formado por 6 capítulos que descrevem os diversos conceitos relacionados a Algoritmo Genético, às Máquinas de Vetores Suporte para regressão e seus principais métodos de treinamento, bem como a proposta desta dissertação, os experimentos e discussão.

A fundamentação teórica para o desenvolvimento desta dissertação com relação à Algoritmos Genéticos é apresentado no Capítulo 2. Esse capítulo apresenta detalhadamente todos os aspectos inerentes a Algoritmos Genéticos necessários para a compreensão da proposta desta dissertação.

No Capítulo 3, é apresentado a fundamentação teórica das Máquinas de Vetores Suporte para regressão, conhecidas como Regressores de Vetores Suporte. Nesse capítulo são salientados os mais populares métodos de treinamento que compreendem o estado da arte no treinamento de Regressores de Vetores Suporte. Cada método de treinamento possui uma subseção dedicada ao seu entendimento.

Os aspectos relacionados a proposta de resolução da formulação dual do problema de otimização quadrática inerente à SVRs por meio de Algoritmos Genéticos são apresentados no Capítulo 4. A proposta, denominada Regressor de Vetor Suporte Evolucionário, utiliza Algoritmos Genéticos para treinar SVRs e cuja abordagem é detalhada nesse capítulo.

Em seguida, no Capítulo 5 são apresentados os experimentos realizados para avaliação da proposta, assim como a apresentação da metodologia que foi utilizada. Ainda nesse capítulo, são realizadas as análises dos resultados obtidos. E, por fim, o Capítulo 6 traz as conclusões e considerações finais desta dissertação, assim como as sugestões para trabalhos futuros.

ALGORITMOS GENÉTICOS

Algoritm os Genéticos pertecem ao grupo de modelos computacionais que usam a evolução natural e a genética como metáfora para gerar soluções úteis para problemas de otimização. O objetivo deste capítulo é descrever a fundamentação teórica necessária sobre Algoritmos Genéticos com o intuito de facilitar a compreensão da proposta para a solução do problema de otimização quadrática na sua formulação dual presente em Regressores de Vetor Suporte. Além disso, ressalta-se que boa parte da fundamentação teórica apresentada neste capítulo encontram-se nas obras de Linden (2008) e Sivanandam e Deepa (2008).

2.1 Introdução

A sobrevivência dos mais aptos é um dos pilares da Teoria da Evolução de Charles Darwin, um dos maiores feitos da ciência do século XIX (LINDEN, 2008). A partir da década de 60 do século passado, com o advento dos computadores, os cientistas passaram a aplicar o mesmo princípio para tentar resolver problemas matemáticos difíceis. A ideia é gerar um grande conjunto de possíveis soluções numéricas (no papel do indivíduo), dar origem a novas soluções a partir das atuais (no papel da reprodução e mutação) e eliminar as demais (atualização da população). Deste modo, após algum processamento (gerações), algumas soluções deveriam estar muito próximas de uma resposta ótima do problema.

Algoritmos Genéticos (AGs) são uma família de meta-heurísticas que se baseiam nos princípios de seleção natural e mutação advindas da evolução biológica de Darwin. AGs têm por base também os conceitos de reprodução propostos por Gregor Mendel, para fins de obtenção de novos indivíduos (filhos) a partir de um par de outros indivíduos denominados pais. A proposição de AGs ocorreu em 1975 por John Holland na Universidade de Michigan e sua popularidade tem crescido a partir de então, particularmente para resolver problemas com um vasto espaço de busca de possíveis soluções (MITCHELL, 1998).

AGs são bastante utilizados para gerar soluções úteis para problemas de busca e otimização.

O uso bastante difundido desta meta-heurística justifica-se pelas vantagens das mesmas sobre uma grande quantidade de métodos clássicos de otimização. Suas características permitem que problemas possam ser resolvidos sem a suposição de linearidade, diferenciabilidade, continuidade ou convexidade da função objetivo (OLIVEIRA; BEZERRA; ROCHA NETO, 2015). Além disso, quando comparados a outros métodos da área de Computação Evolutiva (CE), como: Programação Evolutiva (FOGEL; OWENS; WALSH, 1966), Estratégias Evolutivas (RECHENBERG, 1973) e Programação Genética (KOZA, 1992, 1994), Algoritmos Genéticos se destacam pela flexibilidade, simplicidade de implementação e eficácia na solução encontrada (CARDOSO, 1998; SIVAPATHASEKARAN *et al.*, 2010).

2.2 Visão geral

Em Algoritmos Genéticos a evolução diz respeito a um processo onde as espécies, consideradas candidatas à solução do problema, competem pela sobrevivência e passam pelo processo de reprodução e avaliação, onde os mais aptos tendem a sobreviver (LINDEN, 2008). Como AGs são inspirados na evolução biológica, sua terminologia faz analogia com os termos da genética, a saber (MITCHELL, 1998):

- **Genes** são as características que representam os indivíduos;
- **Cromossomos** são compostos por um aglomerado de genes que juntos representam uma parte da solução ou a própria solução do problema de busca. Um cromossomo é formado por N genes que possuem valor e posição;
- **Alelo** é o valor atribuído ao gene;
- **Locus** é a posição do gene no cromossomo;
- **Genótipo** é a estrutura da solução, é composto por um ou mais cromossomos;
- **Fenótipo** é a decodificação da estrutura do genótipo;
- **Indivíduo** representa um estado do espaço de busca. Quando o indivíduo tem apenas um cromossomo, cromossomo e indivíduo são tratados indistintamente;
- **Ambiente** é o problema, ou seja, o espaço de busca em que se deseja encontrar um estado que maximize uma determinada função objetivo (ou minimize uma determinada função custo); e

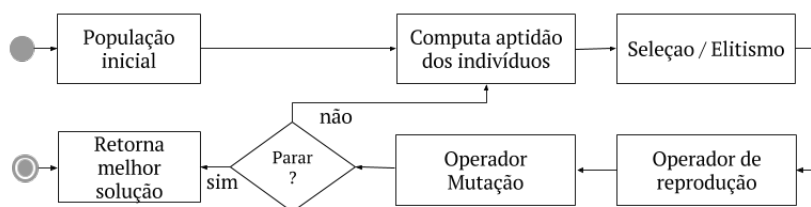
- **Aptidão (*Fitness*)** é o grau de aptidão de um indivíduo ao ambiente, ou seja, o valor obtido da função de aptidão.

Da mesma maneira que ocorre na natureza, a informação dos indivíduos é codificada em cromossomos e a reprodução faz com que a população evolua (explore novas configurações ou estados de busca). A mutação cria diversidade nos indivíduos alterando aleatoriamente os genes, permitindo que novos estados do espaço de busca sejam obtidos, inclusive aqueles que não surgiriam por meio da reprodução.

A execução tradicional de AGs segue os passos apresentados na Figura 1. No processo de evolução em AGs, após a geração do conjunto inicial de soluções (população inicial), os indivíduos são selecionados, com base em algum critério, dois-a-dois para recombinação de seus materiais genéticos. Estes indivíduos resultantes podem sofrer uma mutação de acordo com uma probabilidade de mutação especificada. A nova população obtida é submetida a um processo de seleção natural, por exemplo, a seleção por roleta, torneio ou amostragem universal estocástica (GOLDBERG; HOLLAND, 1988).

A seleção natural favorece a sobrevivência dos mais aptos (as melhores soluções), provendo então a base para um novo ciclo evolucionário (geração). A aptidão de um indivíduo é verificada por uma função, chamada de função de aptidão (*fitness function*). Esta função quantifica o quão boa uma solução é para o problema abordado. Neste contexto, pode acontecer de uma geração ser pior em média que a anterior, ainda que não seja muito comum, mas esta, por sua vez, pode possuir um melhor indivíduo que as gerações anteriores. Para superar e adequar os AGs a este tipo de problema, costuma-se adotar a estratégia de elitismo, na qual os indivíduos mais aptos da geração anterior permanecem nas gerações subsequentes.

Figura 1: Execução de um Algoritmo Genético



Fonte: autor.

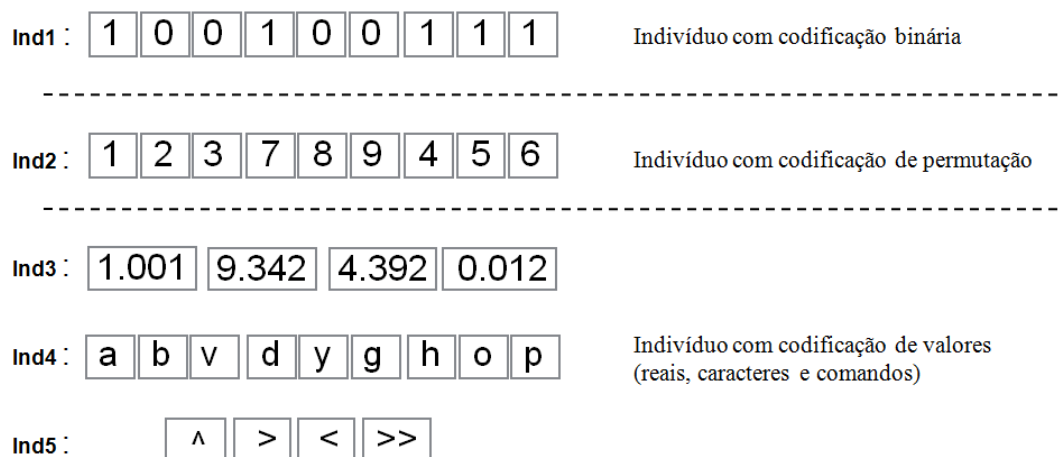
Após a formação de uma geração, os critérios de convergência são verificados. Geralmente procura-se algum indivíduo naquela geração que seja bom o suficiente para ser usado como uma solução para o problema ou se uma determinada condição previamente estabelecida aconteceu. Se o critério de parada não for alcançado, o processo se repete a partir da etapa de seleção até que algum critério de parada seja atendido.

2.3 Representação genética

Um dos principais aspectos em AGs é a representação do problema de maneira que nesta representação seja possível resolver o problema. De modo geral, a representação consiste em uma maneira de traduzir a informação do problema em uma maneira viável para o computador e para o AG.

Para muitos problemas, a representação binária (0 e 1) é suficiente (HOLLAND, 1975), porém com o crescimento de aplicações de AGs e a complexidade dos problemas, abriu-se espaço para outros tipos de representações ou codificações. As principais e mais utilizadas são a binária, inteira e real, mas podem ser também do tipo caractere. As codificações com números inteiros podem ser binárias ou do tipo permutação onde os valores não podem se repetir (SIVANANDAM; DEEPA, 2008). Na Figura 2 são ilustrados exemplos de codificação de indivíduos em Algoritmos Genéticos.

Figura 2: Exemplos de diferentes codificações de indivíduos em Algoritmos Genéticos.



Fonte: autor.

Quanto mais adequada a representação, maior a qualidade dos resultados obtidos. Koza (1992) e Linden (2008) citam algumas características que devem ser observadas na escolha da representação, a saber: (i) simplicidade, a representação deve ser a mais simples possível; (ii) completude, se é possível representar todos os fenótipos e se as soluções podem ser decodificadas; (iii) coerência, se é possível gerar um indivíduo que não pertence ao espectro de soluções do problema, ou seja, soluções inválidas, se houverem, elas não devem ter uma representação. Alternativamente, as soluções inválidas podem ser penalizadas, ou seja, receber valores que significam baixa aptidão, ou ainda removidas ou corrigidas (MICHALEWICZ; JANIOW, 1991).

2.4 População

A população é um conjunto de indivíduos candidatos a participarem do processo de seleção, reprodução e mutação. Geralmente a população inicial é gerada aleatoriamente fornecendo maior biodiversidade e abrangência do espaço de busca. Os operadores de inicialização mais comuns são (GOLDBERG; HOLLAND, 1988):

- **Iniciação heurística:** os indivíduos são criados a partir de heurísticas. No entanto, há a possibilidade de gerar indivíduos semelhantes prejudicando a diversidade da população, para este caso, adotam-se métodos aleatórios em conjunto com heurísticas;
- **Iniciação aleatória não uniforme:** um conjunto de alelos tendem a ser escolhidos com maior frequência;
- **Iniciação aleatória com dope:** indivíduos mais aptos são inseridos na população gerada aleatoriamente. No entanto, há o risco destes indivíduos dominarem o processo de evolução acarretando o problema da convergência prematura (problema onde os filhos são muito semelhantes aos pais ou até mesmo estagnação das evoluções);
- **Iniciação aleatória uniforme:** o gene recebe um valor sorteado aleatoriamente de forma uniforme de um conjunto de alelos.

O tamanho da população é uma das mais importantes escolhas que deve ser levada em consideração ao utilizar AGs, pois uma população pequena pode gerar soluções ruins em muitas aplicações e tendem a convergir rapidamente. Se for muito grande, há um desperdício de recursos computacionais, pois o tempo de espera para alcançar indivíduos mais aptos pode ser muito longo (MICHALEWICZ, 1996). Na prática, um tamanho de população em torno de 100 indivíduos é bastante comum. Entretanto, este tamanho pode ser alterado de acordo com o processamento e memória disponíveis a fim de melhorar a qualidade do resultado a ser atingido (SIVANANDAM; DEEPA, 2008).

2.5 Operadores Genéticos

Operadores genéticos estendem a busca até atingir um resultado satisfatório. Estes transformam a população através de gerações mantendo características boas adquiridas das gerações anteriores. A cada geração os resultados obtidos, em geral, são melhores que os das anteriores.

Operadores genéticos são classificados em operadores de seleção, reprodução e mutação. A escolha destes operadores também é importante, uma vez que há uma variedade deles e cada um com suas características. De uma maneira geral, a escolha depende do problema a ser atacado, pois alguns operadores são mais adequados do que outros para a obtenção de soluções aproximadamente ótimas.

2.5.1 Operadores de Seleção

A etapa da seleção descrita na Figura 1 é responsável por escolher quais indivíduos devem ser utilizados na geração de novos descendentes. Para isto, é necessário a escolha de um método de seleção, pois uma má escolha de reprodutores pode gerar resultados indesejáveis. Em Goldberg e Holland (1988) são citados métodos de seleção, sendo os mais conhecidos os descritos a seguir:

- **Roleta:** cada indivíduo recebe na roleta uma fatia correspondente ao seu valor de aptidão (*fitness*), ou seja, indivíduos mais aptos recebem uma porção maior da roleta, tendo portanto maior probabilidade de serem selecionados e os menos aptos uma porção menor. A roleta é girada um determinado número de vezes, de tal maneira que a cada giro um indivíduo é selecionado. O genoma dos indivíduos sorteados podem participar da próxima geração;
- **Amostragem Universal Estocástica (*Stochastic Universal Sampling* - SUS):** semelhante à roleta, mas para selecionar k indivíduos utiliza k agulhas igualmente espaçadas, girando-as em conjunto uma só vez; e
- **Torneio:** k indivíduos são sorteados e o mais apto vence. Logo, todos que vencem o torneio são os selecionados pelo método.

2.5.2 Elitismo

Os indivíduos que passam pelos operadores de reprodução e mutação são substituídos pelos novos indivíduos gerados, pode ocorrer destes indivíduos serem ótimos indivíduos. Para evitar que ótimas soluções se percam após reprodução e mutação, o elitismo mantém os melhores indivíduos da geração atual na seguinte ou substitui os piores indivíduos da geração atual pelos melhores da anterior.

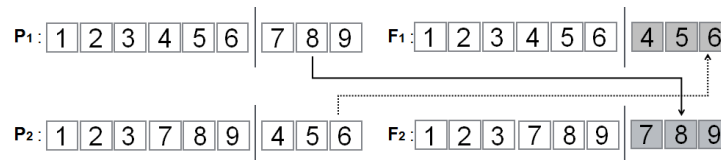
2.5.3 Operadores de Reprodução

Após a seleção dos indivíduos reprodutores, o próximo operador é o de reprodução. Esta, muitas vezes denominada *crossover*, é uma etapa também inspirada na natureza e tem como objetivo inserir novas soluções na população através da mistura de informação gênica contida nos cromossomos dos indivíduos pais para gerar os filhos. Este operador gera novos indivíduos que formarão a próxima geração e atua com uma probabilidade dada pela taxa de cruzamento P_c , ou seja, o operador é aplicado a uma parte da geração atual.

Assim, considerando as diversas combinações possíveis das características dos pais, um número bastante elevado de soluções podem ser geradas. Em AGs, as implementações de *crossover*, em geral, produzem dois novos indivíduos a partir de outros dois. Os operadores mais sofisticados e utilizados para realizar esta combinação são os seguintes:

- **Operador *crossover* de um ponto:** um ponto de corte é escolhido entre os genes do indivíduo (HOLLAND, 1975). Considere P_1 e P_2 os pais e F_1 e F_2 os filhos. F_1 recebe as informações genéticas anteriores ao ponto de corte em P_1 e posteriores ao ponto de corte em P_2 . F_2 recebe as informações genéticas anteriores ao ponto de corte em P_2 e posteriores a ponto de corte em P_1 , veja Figura 3;

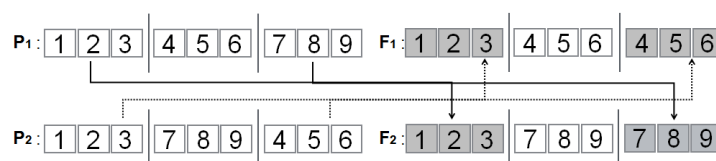
Figura 3: Funcionamento do *crossover* de 1 ponto



Fonte: autor.

- **Operador *crossover* multi-ponto:** semelhante ao operador anterior, neste caso, o sorteio de um número n determina a quantidade de pontos de corte que serão utilizados (GOLDBERG; HOLLAND, 1988). Veja um exemplo com $n = 2$ na Figura 4;

Figura 4: Funcionamento do *crossover* multi-ponto,

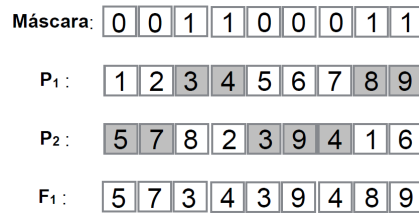


Fonte: autor.

- **Operador *crossover* uniforme:** é gerado uma máscara aleatoriamente com valores 0 e 1 do mesmo comprimento dos pais selecionados P_1 e P_2 . Este operador gera um filho F_1 e

os genes são obtidos percorrendo as posições da máscara, caso o valor da posição da máscara seja 1, o gene de P_1 referente a posição da máscara é copiado para F_1 , se o valor for 0 então o gene de P_2 é copiado para F_1 (SYSWERDA, 1989). O funcionamento do operador pode ser observado na Figura 5;

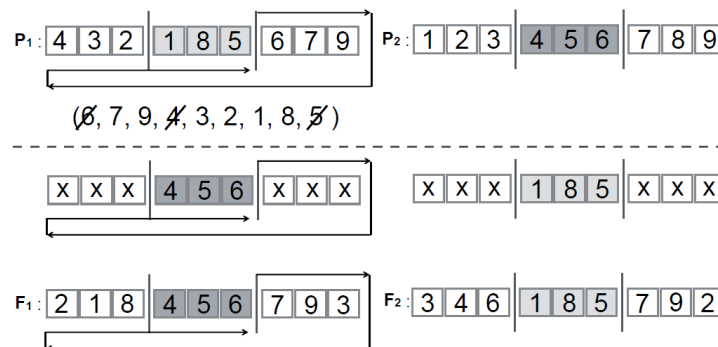
Figura 5: Funcionamento do *crossover* uniforme



Fonte: autor.

- **Operador OX (*ordered crossover*):** operador de permutação para indivíduos compostos de genes que não podem ter seus valores repetidos. Neste operador dois pontos de cortes são escolhidos e posicionados aleatoriamente (DAVIS, 1985). Observe a Figura 6. Inicialmente, os pontos de cortes e suas posições são escolhidas aleatoriamente. A partir disto, F_1 recebe a subsequência de informações genéticas entre os pontos de corte de P_2 (4,5,6) e F_2 a subsequência entre os pontos de corte de P_1 (1,8,5). Em seguida, uma sequência de valores de genes de P_1 é obtida iniciando-a em uma das extremidades dos pontos de corte e seguindo uma direção, por exemplo, na figura a sequência é dada por $s_1 = 6, 7, 9, 4, 3, 2, 1, 8, 5$. Desta sequência s_1 , removem-se os valores já atribuídos a F_1 (4,5,6), assim $s_1 = 7, 9, 3, 2, 1, 8$. Seguindo a mesma ordem e iniciando na mesma extremidade, os valores de s_1 são atribuídos aos genes de F_1 . Da mesma maneira ocorre em F_2 com a sequência de valores de genes obtida da mesma maneira a partir de P_2 ($s_2 = 7, 8, 9, 1, 2, 3, 4, 5, 6$);

Figura 6: Funcionamento do *crossover* OX



Fonte: autor.

- **Operador aritmético** : operador de cruzamento aritmético (MÜHLENBEIN; SCHLIERKAMP-VOOSEN, 1993) em que dados dois pais, os filhos são resultados da combinação linear dos materiais genéticos dos pais. Tal comportamento resulta em filhos que são factíveis no que diz respeito a restrições lineares e limites da representação. Seja P_1 e P_2 dois indivíduos, os filhos F_1 e F_2 , são produzidos como segue:

$$F_1 = \gamma \cdot P_1 + (1 - \gamma) \cdot P_2, \quad (2.1)$$

$$F_2 = \gamma \cdot P_2 + (1 - \gamma) \cdot P_1 \quad (2.2)$$

em que $\gamma \in [0, 1]$. Existem variantes deste operador em que cada par possui um valor específico de γ , visando dar mais variabilidade ao processo de cruzamento.

2.5.4 Operadores de Mutação

Os operadores de mutação são responsáveis por manter a diversidade genética na população e, conseqüentemente, explorar novos espaços de busca. Esta etapa ocorre após a reprodução, modificando aleatoriamente um ou mais genes de um cromossomo. A probabilidade de ocorrência em um gene é dada pela taxa de mutação. O operador percorre cada gene do cromossomo gerando um evento de probabilidade Pm e se este evento ocorrer, o valor do gene é alterado. Usualmente as taxas de mutação recebem valores variando de 0 a 1, sendo os mais utilizados 0,001 e 0,01 (BÄCK; FOGEL; MICHALEWICZ, 1997). É importante atentar que quando a taxa de mutação é muito alta, a busca em AGs pode se tornar essencialmente aleatória. Os principais operadores de mutação são, os seguintes (GOLDBERG; HOLLAND, 1988), (DAVIS, 1991)

- **Mutação aleatória (*flip mutation*)**: cada gene mutado recebe um valor sorteado pertencente ao alfabeto válido¹;
- **Mutação por troca (*swap mutation*)**: são sorteados n pares de genes e os pares de genes trocam os valores entre si;
- **Mutação *creep***: um valor aleatório é somado ou subtraído do valor do gene.

¹Entenda como alfabeto válido, o universo no qual os valores dos genes estão contidos, inteiros, string, binários, etc

2.6 Função de aptidão (*fitness*)

A função de aptidão deve ser escolhida criteriosamente, pois ela é responsável por retornar o grau de aptidão de um indivíduo, avaliando sua qualidade enquanto solução do problema. Este valor é utilizado nas etapas de seleção e reprodução. Desta forma, quando mal elaborada, pode acarretar na escolha de reprodutores ruins e gerar resultados insatisfatórios. Toda a informação relativa ao problema deve ser embutida nesta função, assim como seus objetivos de qualidade (LINDEN, 2008).

A função de aptidão, $F(\text{indivíduo})$, em geral, atribui um valor de aptidão ao indivíduo onde, geralmente, valores altos de aptidão indicam que o indivíduo representa uma solução melhor que os demais. Desta forma, a medida que os indivíduos violam o conjunto de restrições que estão incluídas na função de aptidão, eles são penalizados e seu valor de aptidão reduzido. A função F pode ser em ambos os termos de minimização ou maximização, depende das características e modelagem do problema aplicado. Vale ressaltar que, um valor de aptidão igual à 0 ou negativo, pode influenciar no operador de seleção, quando utilizado o método da roleta, onde o indivíduo obtém uma porção da mesma (LINDEN, 2008).

2.7 Critérios de convergência

Os critérios de convergência (ou término) encerram a busca em AGs. Critérios podem ser combinados para prover um mecanismo capaz de verificar uma possível estagnação do processo evolutivo ou alternativamente, para verificar uma restrição em termos de processamento computacional. Alguns critérios de convergência são apresentados a seguir:

1. **Número máximo de gerações:** o processo de busca é finalizado com base no número de gerações já realizadas.
2. **Tempo máximo de execução:** o processo de busca é finalizado com base no tempo já transcorrido. Lembrando que se o número máximo de gerações for alcançado primeiro, AGs tem sua execução encerrada também.
3. **Sem mudança no melhor valor de aptidão:** a busca é encerrada quando não há mudança no melhor valor de aptidão encontrado durante um determinado número de gerações.
4. **Sem mudança na média da função objetivo de consecutivas gerações:** o processo de busca é interrompido quando não há uma melhoria na média da função objetivo para um número definido de gerações consecutivas.

5. **Sem mudança na média da função objetivo de consecutivas gerações durante um intervalo de tempo:** o processo de busca é finalizado quando não há uma melhoria na média da função objetivo de consecutivas gerações durante um intervalo de tempo definido.

2.8 Conclusão

Esse capítulo expõe os principais aspectos relacionados à Algoritmos Genéticos. As características de AGs se adequam ao problema de programação quadrática inerente a SVR e vários aspectos relacionados à AGs os tornam alternativas interessantes para esta aplicação. Devido as suas características e através de uma modelagem bem projetada, AGs podem ser aplicados a problemas de otimização, sendo capazes de gerar soluções tão boas quanto métodos matemáticos clássicos que dependem da hipótese de linearidade, diferenciabilidade, continuidade ou convexidade da função objetivo.

MÁQUINAS DE VETORES SUPORTE PARA REGRESSÃO

Neste capítulo, são apresentados em detalhes os aspectos relacionados as Máquinas de Vetores Suporte para regressão e os métodos de treinamento mais utilizados na literatura, o de Programação Quadrática, o algoritmo de otimização sequencial mínima (*Sequential Minimal Optimization*, SMO) e o algoritmo único de dados iterativo (*Iterative Single Data Algorithm*, ISDA). Além disso, uma solução baseada em metaheurística também é apresentada. Sempre que possível, comparações entre SVMs para classificação e regressão são realizadas a fim de facilitar a compreensão sobre o funcionamento de máquinas de vetores suporte para regressão.

3.1 Introdução

Máquinas de Vetores Suporte (SVM) foram desenvolvidas por Vapnik (VAPNIK, 1995) e ganharam popularidade devido as suas características de desempenho. A sua popularidade pode ser justificada pelo forte arcabouço teórico utilizado em sua formulação. Em contraste com as redes neurais artificiais, as SVMs não precisam determinar a topologia da rede com antecedência. Além disso, as máquinas de vetores suporte tem a capacidade de resolver problemas de alta dimensionalidade (WANG *et al.*, 2003).

O principal motivo que faz com que as SVMs se destaquem é o fato de adotarem o princípio de minimização do risco estrutural que tem se mostrado superior à minimização do risco empírico aplicado por redes neurais convencionais (GUNN; BROWN; BOSSLEY, 1997). O erro empírico está relacionado ao erro no conjunto de treinamento, ao passo que o risco estrutural está relacionado ao erro sobre aplicar o modelo sobre padrões desconhecidos, ou seja, o conjunto de teste. A minimização do erro estrutural aumenta a capacidade de classificação/regressão do modelo sobre o conjunto de teste, no entanto, minimizar o risco empírico não implica em necessariamente reduzir o erro de generalização (HUANG;

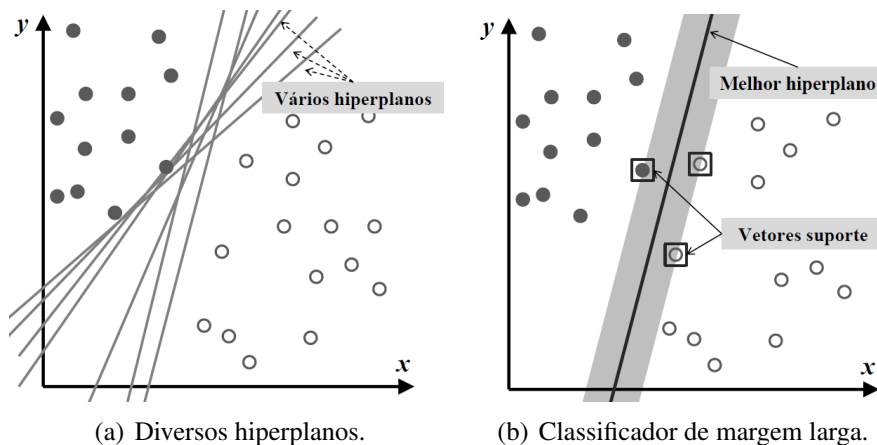
KECMAN; KOPRIVA, 2006).

Originalmente, as SVMs foram formuladas para problemas de classificação de padrões, denominadas como classificadores de vetores suporte (*Support Vector Classifiers*, SVCs). Porém, com a utilização da função de perda ε -insensitive, as SVMs passaram a ser aplicadas também à problemas de regressão e predição de séries temporais (VAPNIK, 1995; VAPNIK; GOLOWICH; SMOLA, 1997). Quando aplicadas à problemas de regressão são denominadas como regressores de vetores suporte (*Support Vector Regressors*, SVR).

3.2 Classificadores de Vetores Suporte

Quando se deseja categorizar padrões com classificadores lineares, existem muitos hiperplanos possíveis que são capazes de separar os dados corretamente. No entanto, somente um deles é capaz de alcançar a separação máxima dos dados. A razão que justifica a busca por esse hiperplano de separação máxima, é que se for utilizado um hiperplano qualquer para classificar, este pode estar mais próximo de um conjunto de dados comparado aos demais. Tal comportamento não é desejado. A escolha pelo hiperplano que maximiza a margem resulta em melhor desempenho e evita mínimos locais. Se for cometido um erro pequeno na localização, a possibilidade é mínima de erro de classificação (JAKKULA, 2006). A Figura 7 apresenta a situação exemplificada anteriormente com um classificador de vetor suporte, onde na Figura 7 (a) são apresentados dois conjuntos de dados e entre eles diversos hiperplanos possíveis. A Figura 7 (b) apresenta o melhor hiperplano, aquele que maximiza a margem representada em cinza e os padrões que estão próximos ao hiperplano são os vetores suporte (JAKKULA, 2006).

Figura 7: Exemplo de escolha do hiperplano com SVM linear.



Fonte: adaptado de (JAKKULA, 2006).

Considere então um conjunto de treinamento $\mathbf{X} = \{\mathbf{x}_i, y_i\}_{i=1}^l$ em que $\mathbf{x}_i \in \mathbb{R}^p$ é o vetor de

entrada, p é a dimensão dos padrões e $y_i \in \{-1, +1\}$ é a classe do i -ésimo padrão. Deseja-se encontrar uma superfície de decisão gerada por um hiperplano em que

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b = 0, \quad b \in \mathbb{R}, \quad (3.1)$$

onde $\mathbf{w}$ é o vetor de pesos e b o viés.

O que se procura, como explicado anteriormente, é um hiperplano que separe corretamente o conjunto de dados, ou seja, os valores para $\mathbf{w}$ e b devem agrupar os padrões de uma classe em um lado do hiperplano e os da outra classe do outro lado do hiperplano. Nas abordagens envolvendo SVMs, o risco estrutural é minimizado através do conceito de maximização da distância, que é a margem larga, dada como a menor distância entre o hiperplano e algum padrão do conjunto de dados. Os padrões próximos ao hiperplano são os vetores suporte, e esses hiperplanos de suporte delimitam a margem, que é a região sombreada na Figura 7. O conjunto de hiperplanos possíveis para esta tarefa seguem as regras (MAVROFORAKIS; THEODORIDIS, 2006)

$$\mathbf{w}^T \mathbf{x}_i \geq 0, \quad \text{se } y_i = +1, \quad (3.2)$$

$$\mathbf{w}^T \mathbf{x}_i \leq 0, \quad \text{se } y_i = -1. \quad (3.3)$$

Tais restrições podem ser reescritas como (BEN-HUR; WESTON, 2010)

$$y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1, i = 1, \dots, l. \quad (3.4)$$

Para encontrar o hiperplano de margem larga, deve-se minimizar $\frac{1}{2} \|\mathbf{w}\|^2$. Logo, o treinamento de um SVC é dado pela resolução do seu problema primal

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2, \quad (3.5)$$

sujeito a

$$y_i(\mathbf{w}^T \mathbf{x}_i - b) \geq 1, i = 1, \dots, l. \quad (3.6)$$

Através do método de multiplicadores de Lagrange, temos que a Equação (3.5) pode ser reescrita como o lagrangiano

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^l \alpha_i (y_i(\mathbf{w}^T \mathbf{x}_i - b) - 1), \quad (3.7)$$

em que $\{\alpha_i\}_{i=1}^l$ é o conjunto de multiplicadores de Lagrange (VAPNIK, 1995). A solução do problema é dada pelo ponto de sela da função lagrangiana apresentada na Equação (3.7),

minimizando-a em relação a $\mathbf{w}$ e b , e maximizando-a em relação a $\boldsymbol{\alpha}$. Utilizando as condições de otimalidade *Karush-Kuhn-Tucker* (KKT) (VAPNIK, 1995), o problema pode então ser apresentado como segue

$$\max \left\{ \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right\}, \quad (3.8)$$

sujeito a

$$\sum_{i=1}^l \alpha_i y_i = 0, \alpha_i \geq 0 \quad (3.9)$$

Este problema apresentado é conhecido como SVC de margem rígida, visto que o mesmo só poder ser aplicado para problemas linearmente separáveis, o que não reflete a realidade visto que na grande maioria dos problemas existentes, há sobreposição de dados entre as classes. Desta forma, certamente não há um hiperplano que seja capaz de separar totalmente os padrões sem a ocorrência de erros de classificação (BISHOP, 2007).

Por este motivo, torna-se necessária a adoção de artifícios para adequação do SVC à problemas com sobreposição de dados, onde o objetivo é tornar a margem flexível para permitir que alguns padrões do treinamento sejam incorretamente classificados (CORTES; VAPNIK, 1995). Esta abordagem resultou no SVC de margem flexível, em que o problema primal é definido como segue

$$\min_{\mathbf{w}, \boldsymbol{\xi}} \left\{ \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^l \xi_i \right\} \quad (3.10)$$

sujeito a

$$y_i[(\mathbf{w}^T \mathbf{x}_i) + b] \geq 1 - \xi_i \quad (3.11)$$

$$\xi_i \geq 0 \quad (3.12)$$

em que $\{\xi_i\}_{i=1}^l$ é o conjunto de variáveis de folga que permitem uma quantidade de erros para classificação, b é o viés e C é o parâmetro responsável pela minimização da complexidade do modelo, estabelecendo assim um equilíbrio entre complexidade do modelo e risco empírico. O parâmetro C permite que a margem seja flexível e ocorra uma certa quantidade de classificações incorretas. A solução do problema na Equação (3.10) é o ponto de sela da função lagrangiana

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^l \xi_i - \sum_{i=1}^l [\alpha_i (y_i (\mathbf{x}_i^T \mathbf{w} + b) - 1 + \xi_i) + \beta_i \xi_i], \quad (3.13)$$

em que $\{\alpha_i\}_{i=1}^l$ e $\{\beta_i\}_{i=1}^l$ são os conjuntos de multiplicadores de Lagrange. Para a resolução

deste problema, a função deve ser minimizada em relação a $\mathbf{w}$, b e ξ e maximizada em relação a α e β . A fim de encontrar a solução para o problema descrito na Equação (3.13) é necessário calcular as seguintes condições de otimalidade

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial \mathbf{w}} = 0, \quad (3.14)$$

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial b} = 0 \quad \text{e} \quad (3.15)$$

$$\frac{\partial L(\mathbf{w}, b, \xi, \alpha, \beta)}{\partial \xi} = 0 \quad (3.16)$$

as quais resultam em

$$\mathbf{w} = \sum_{i=1}^l \alpha_i y_i \mathbf{x}_i, \quad (3.17)$$

$$\sum_{i=1}^l \alpha_i y_i = 0 \quad (3.18)$$

e

$$C = \alpha_i + \beta_i, \quad (3.19)$$

respectivamente. A partir disso, essas expressões são introduzidas na Equação (3.13) e o problema de otimização dual do SVM pode ser representado como segue

$$\max \left\{ \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \alpha_i \alpha_j y_i d_j \mathbf{x}_i^T \mathbf{x}_j \right\}, \quad (3.20)$$

sujeito a

$$\sum_{i=1}^l \alpha_i y_i = 0 \quad (3.21)$$

e

$$0 \leq \alpha_i \leq C. \quad (3.22)$$

Depois da resolução do problema e ter encontrado os multiplicadores de Lagrange e o viés, a saída pode então ser computada com base na função de decisão

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i=1}^l \alpha_i y_i \mathbf{x}^T \mathbf{x}_i + b \right). \quad (3.23)$$

Se os dados são linearmente separáveis, um hiperplano pode ser traçado para separá-los.

No entanto, geralmente o conjunto de treinamento não é linearmente separável no espaço de entrada. Para que seja possível obter um problema linear a partir de um não-linear, é necessário realizar uma transformação não-linear nos dados de entrada $\mathbf{x}_i \in \mathbb{R}^N$ levando-os para um espaço de alta dimensionalidade, o espaço de características $\Phi(\mathbf{x}) \in \mathbb{R}^M$, tal que $M > N$. Neste novo mapeamento os dados são linearmente separáveis (COVER, 1965; ROCHA NETO, 2011).

Para que seja evitado o uso explícito deste espaço de características de alta dimensionalidade, temos o truque de *kernel* $K(\mathbf{x}, \mathbf{x}_i)$. Por meio das funções de *kernel* (linear, polinomial, Gaussiano, etc) é possível trabalhar indiretamente no espaço de características, porém em um outro espaço, o Espaço *Kernel*. Desta forma, a obtenção de um hiperplano de separação dos dados neste espaço de características se torna possível (ROCHA NETO, 2011; COVER, 1965). Na Equação (3.20) e (3.23) a função de *kernel* é inserida através da substituição de $\mathbf{x}^T \mathbf{x}_i$ por $K(\mathbf{x}, \mathbf{x}_i)$.

3.3 Regressores de Vetores Suporte

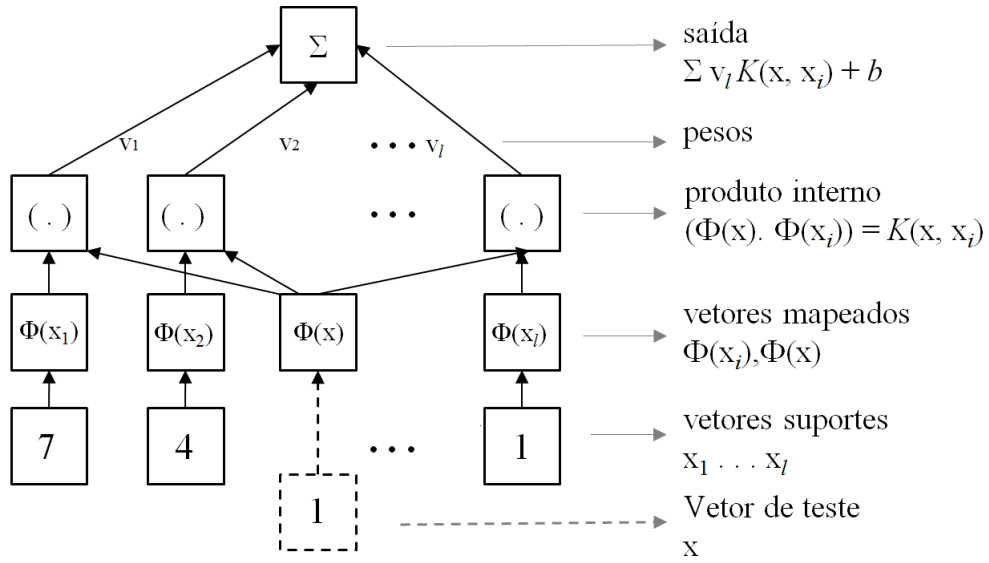
O método de vetor suporte é uma ferramenta universal para a resolução de problemas de estimação de funções n-dimensionais. Inicialmente, SVMs foram projetadas para problemas de classificação, nos quais um subconjunto dos dados de treinamento é selecionado, que são os vetores suporte, para encontrar uma regra de decisão com uma boa capacidade de generalização. A separação ótima desses vetores suporte é equivalente a separação ótima do conjunto de dados como um todo. A partir daí, um novo método para representação de funções de decisão foi desenvolvido, onde as funções de decisão são uma expansão linear cujo os elementos são funções não-lineares parametrizadas pelos vetores suporte¹. Essa representação é útil para espaços de entrada de alta dimensionalidade, nos quais o número de parâmetros livres nesta representação é igual ao número de vetores suporte, mas não depende da dimensionalidade do espaço de entrada. Assim, com esse novo tipo de representação que permite expandir funções de alta dimensionalidade utilizando como base um pequeno conjunto de vetores suporte, SVMs foram estendidas para estimação e aproximação de funções, os chamados regressores de vetores suporte (*Support Vector Regressors*, SVR) (VAPNIK; GOLOWICH; SMOLA, 1997).

Os componentes básicos para estimação de funções a partir de SVRs podem ser vistos na Figura 8. Esta figura apresenta uma visão geral dos passos no processo de regressão. Os padrões de entrada $\mathbf{x}$, que são do conjunto de teste, são mapeados pela função Φ para o espaço de características. Os valores do produto interno são computados com os padrões do conjunto de

¹É necessário um vetor suporte para cada elemento do conjunto de dados

treinamento também sob o mapeamento da função Φ . Este passo corresponde a chamada função de kernel $K(\mathbf{x}_i, \mathbf{x})$. Por último, o resultado dos produtos internos são ponderados pelos pesos $v_i = \alpha_i - \alpha_i^*$, em que α_i e α_i^* correspondem aos multiplicadores de Lagrange, e em seguida somados. O resultado da soma do termo que contém o somatório mais a constante viés b é a saída da predição. Este processo é bem similar a regressão em redes neurais, com uma diferença, de que aqui os pesos da camada de entrada são um subconjunto do conjunto de treinamento (SMOLA; SCHOLKOPF, 2004).

Figura 8: Arquitetura de um regressor de vetor suporte.



Fonte: adaptado de (SMOLA; SCHOLKOPF, 2004).

Considere um conjunto de treinamento $\mathbf{X} = \{\mathbf{x}_i, y_i\}_{i=1}^l$, tal que $\mathbf{x}_i \in \mathbb{R}^d$ são os padrões de entrada e $y_i \in \mathbb{R}$ é a variável dependente correspondente. O objetivo é encontrar um hiperplano de regressão dado pela função

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b, \quad b \in \mathbb{R}, \quad (3.24)$$

que tenha no máximo um desvio ε da variável dependente y_i .

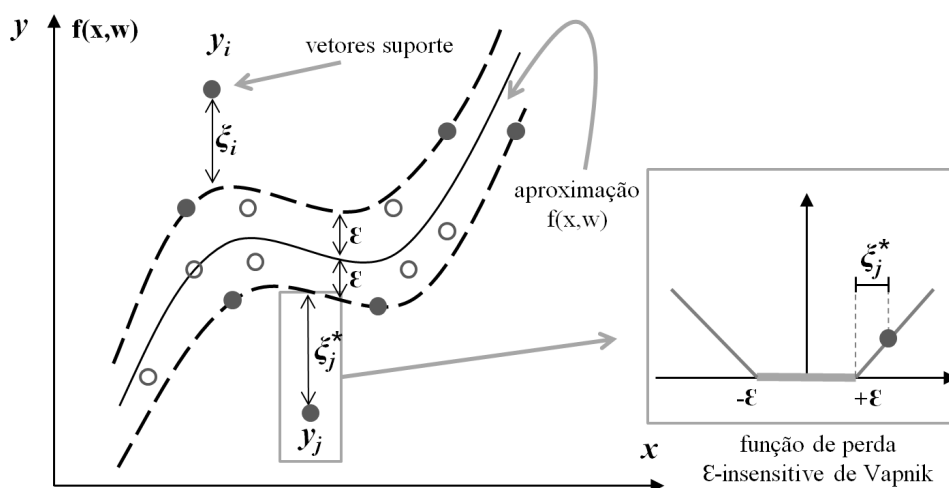
No caso de um SVR, é medido o erro de aproximação ao invés da margem utilizada em um SVC. A diferença mais importante é que essa medida do erro de aproximação corresponde a lidar com uma função de perda $|\xi_i|_\varepsilon$, a ε -insensitive de Vapnik (VAPNIK; GOLOWICH; SMOLA, 1997) dada como segue

$$|\xi_i|_\varepsilon = \begin{cases} 0, & \text{se } |\xi_i| \leq \varepsilon, \\ |\xi_i| - \varepsilon & \text{caso o contrário.} \end{cases} \quad (3.25)$$

Esse valor de perda $|\xi_i|_\varepsilon = |y - f(\mathbf{x})|$ será igual a zero se a diferença entre o valor da predição

e o esperado y_i for menor que ε e essa diferença é utilizada como o erro. A função de perda ε -insensitive é representada por um tubo como apresentado na Figura 9, onde o tubo é dado pelas linhas tracejadas. Logo, se os valores da predição estiverem dentro do tubo formado pela função ε -insensitive, significa que a perda é zero, ou seja, o erro é zero. Para todos os outros valores da predição que estejam fora do tubo, o valor da perda ou erro é dado pelo valor absoluto da diferença entre o valor da predição e o raio do tubo que é ε , ou seja, $|\xi_i| - \varepsilon$ (HUANG; KECMAN; KOPRIVA, 2006). Ainda na figura, apenas os pontos fora da região tracejada contribuem para o erro. Os padrões preenchidos são vetores suporte e todos os padrões dentro tubo ε não são vetores suporte. Mais adiante as variáveis de folga (ξ_i e ξ_j^*) apresentadas na figura serão explicadas.

Figura 9: Variáveis de folga em SVR e função de perda ε -insensitive.

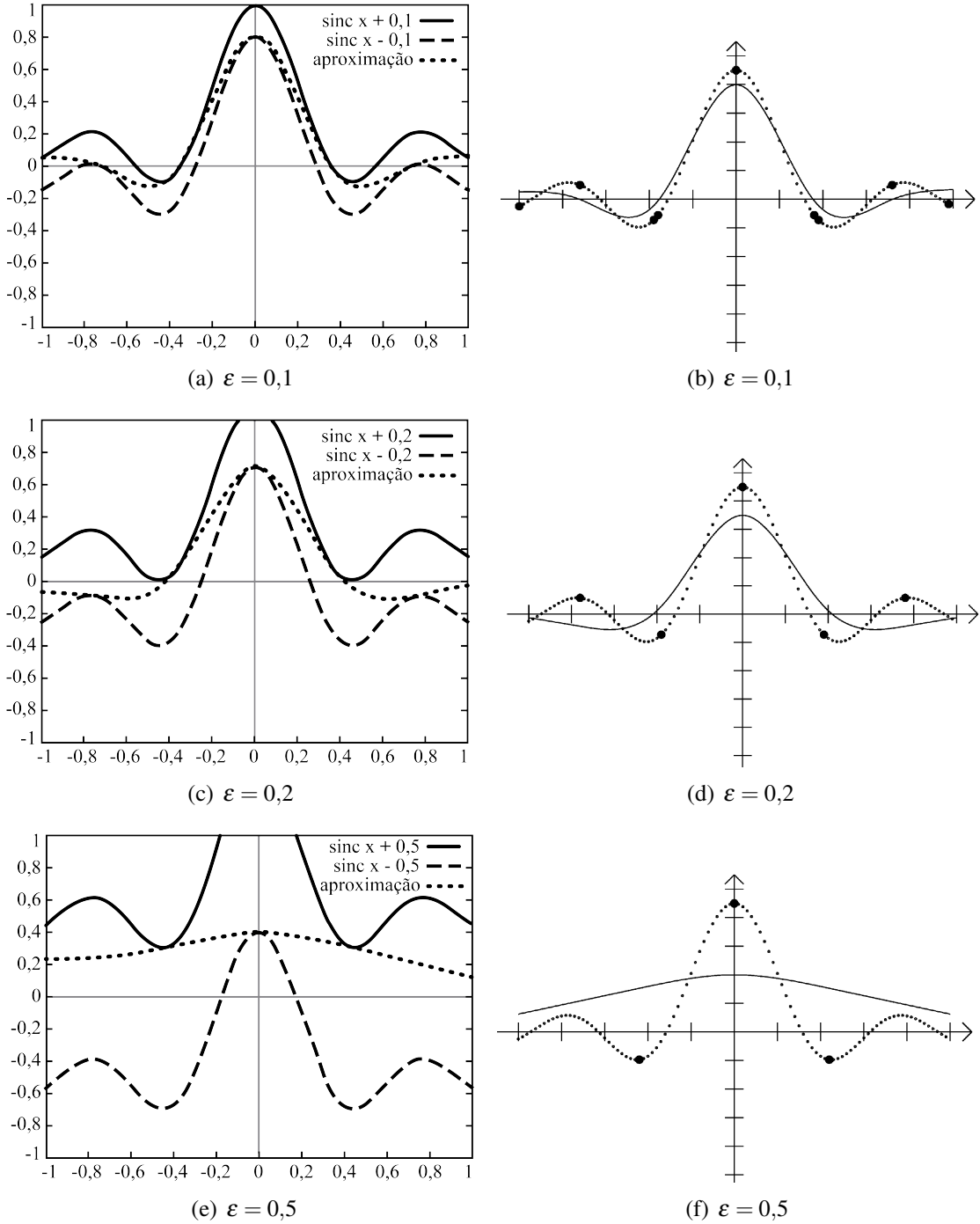


Fonte: adaptado de (HUANG; KECMAN; KOPRIVA, 2006).

Dentre as possíveis funções $f(\mathbf{x})$ que aproximam os dados, um SVR escolhe uma função que os aproxime com uma certa precisão ε . Como pode ser visto na Figura 10, quanto maior o valor associado a ε pior o modelo de aproximação encontrado, uma vez que ε define que somente serão considerados como erro, o valor de perda ou erro que seja maior que ε .

A Figura 10 (a) apresenta o modelo SVR com menor valor de ε , e como pode ser visto, o modelo aproximou a função melhor que na Figura 10 (e). Outra relação importante é que quanto menor a precisão para aproximar os dados, maior a quantidade de vetores suportes necessários, como mostra a Figura 10 (b). Consequentemente, na Figura 10 (f) com ε maior, foram necessários menos vetores suporte. Vale ressaltar que os padrões que não são vetores suporte são redundantes, ou seja, mesmo sem estes padrões no conjunto de treinamento, a mesma aproximação seria alcançada. Desta forma, é necessário armazenar somente os padrões de suporte, os quais são os únicos considerados necessários para a estimação da função.

Figura 10: Aproximação da função sinc x com precisão $\varepsilon \in 0,1; 0,2; 0,5$. Em (a), (c) e (e): a linha sólida superior e a linha tracejada inferior indica o tamanho do tubo ε , a linha pontilhada é $f(\mathbf{x})$. Em (b), (d) e (f): a relação entre a aproximação (linha sólida) e a esparsidade. Os dados são os pontos menores e os vetores suportes os pontos maiores.



Fonte: (SMOLA; SCHOLKOPF, 2004).

O risco empírico $R_{emp}^{\varepsilon}(\mathbf{w}, b) = \frac{1}{l} \sum_{i=1}^l |y_i - \mathbf{w}^T \mathbf{x}_i - b|_{\varepsilon}$ em SVR é medido então pela função de perda ε -insensitive apresentada na Equação (3.25). Da mesma forma como em SVCs, busca-se em SVR minimizar ambos o risco empírico e a norma dos pesos $\|\mathbf{w}\|^2$,

simultaneamente. Assim, a função da Equação (3.24) é encontrada a partir da minimização do risco

$$R = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^l |y_i - f(\mathbf{x}_i)|_{\varepsilon}. \quad (3.26)$$

Tendo em vista a função de perda apresentada na Equação (3.25) e a Figura 9, temos que para todos os dados de treinamento fora do tubo ε ,

$$|y - f(\mathbf{x})| - \varepsilon = \xi \quad \text{para os dados acima do tubo } \varepsilon, \text{ ou} \quad (3.27)$$

$$|y - f(\mathbf{x})| - \varepsilon = \xi^* \quad \text{para os dados abaixo do tubo } \varepsilon, \quad (3.28)$$

sendo assim, minimizar o risco R da Equação (3.26) significa minimizar o risco que segue, o qual representa o problema primal em SVR (VAPNIK, 1995; HUANG; KECMAN; KOPRIVA, 2006)

$$\min_{\mathbf{w}, \xi, \xi^*, b} \left\{ \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^l (\xi_i + \xi_i^*) \right\}, \quad (3.29)$$

$$\text{sujeito a} \quad \begin{cases} y_i - \mathbf{w}^T \mathbf{x}_i - b \leq \varepsilon + \xi_i \\ \mathbf{w}^T \mathbf{x}_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i \geq 0 \\ \xi_i^* \geq 0 \end{cases}, \quad (3.30)$$

onde b é o viés e $\{\xi_i\}_{i=1}^l$ e $\{\xi_i^*\}_{i=1}^l$ são as variáveis de folga que permitem alguns erros. Enquanto que a constante $C > 0$ controla o compromisso da complexidade do modelo apresentado na Equação (3.24) e o quanto de desvios maiores que ε são tolerados. Um valor maior para C penaliza os erros de maior magnitude e força que as variáveis de folga $\{\xi_i\}_{i=1}^l$ e $\{\xi_i^*\}_{i=1}^l$ sejam pequenas.

Similar aos procedimentos aplicados em SVCs, o problema de otimização com restrições

apresentado na Equação (3.29) é solucionado pela construção da função lagrangiana que segue

$$\begin{aligned}
 L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*) &= \frac{1}{2} \mathbf{w}^T \mathbf{w} + C \sum_{i=1}^l (\xi_i + \xi_i^*) \\
 &\quad - \sum_{i=1}^l (\beta_i \xi_i + \beta_i^* \xi_i^*) \\
 &\quad - \sum_{i=1}^l \alpha_i (\mathbf{w}^T \mathbf{x}_i + b - y_i + \varepsilon + \xi_i) \\
 &\quad - \sum_{i=1}^l \alpha_i^* (y_i - \mathbf{w}^T \mathbf{x}_i - b + \varepsilon + \xi_i^*). \tag{3.31}
 \end{aligned}$$

As variáveis primais da função lagrangiana $L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*)$ devem ser minimizadas em relação as variáveis primais $\mathbf{w}$, b , $\boldsymbol{\xi}$ e $\boldsymbol{\xi}^*$ e maximizadas com relação aos multiplicadores de Lagrange $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$, $\boldsymbol{\beta}$ e $\boldsymbol{\beta}^*$. As condições de otimalidade no ponto de sela para esse problema são obtidas colocando todas as derivadas em relação as variáveis do problema primal ($\mathbf{w}, b, \boldsymbol{\xi}$ e $\boldsymbol{\xi}^*$) iguais a zero (HUANG; KECCMAN; KOPRIVA, 2006), obtendo assim, as chamadas condições de otimalidade *Karush-Kuhn-Tucker* (KKT) definidas como segue

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*)}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^l (\alpha_i - \alpha_i^*) \mathbf{x}_i, \tag{3.32}$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*)}{\partial b} = 0 \Rightarrow \sum_{i=1}^l (\alpha_i^* - \alpha_i) = 0, \tag{3.33}$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*)}{\partial \xi_i} = 0 \Rightarrow \beta_i = C - \alpha_i, \tag{3.34}$$

$$\frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\xi}^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\beta}, \boldsymbol{\beta}^*)}{\partial \xi_i^*} = 0 \Rightarrow \beta_i^* = C - \alpha_i^*. \tag{3.35}$$

Assim, a formulação dual é obtida através da substituição das condições de otimalidade na Equação (3.31), chegando assim no problema de maximização dual e na função lagrangiana

$$\max_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} \left\{ -\frac{1}{2} \sum_{i,j=1}^l (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \mathbf{x}_i^T \mathbf{x}_j - \varepsilon \sum_{i=1}^l (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \right\}, \tag{3.36}$$

$$\text{sujeito a } \alpha_i, \alpha_i^* \in [0, C], \text{ e} \tag{3.37}$$

$$\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0. \tag{3.38}$$

Como consequência, após a substituição do valor da Equação (3.32), a Equação (3.24) pode ser

reescrita da seguinte forma

$$f(\mathbf{x}) = \sum_{i=1}^l (\alpha_i - \alpha_i^*) \mathbf{x}_i^T \mathbf{x} + b. \quad (3.39)$$

Note que o problema dual apresentado na Equação (3.36) é expresso em termos de dois multiplicadores de Lagrange, o dobro do problema dado em SVCs. Há então $2l$ variáveis desconhecidas $\{\alpha_i\}_{i=1}^l$ e $\{\alpha_i^*\}_{i=1}^l$ e a matriz Hessiana $\mathbf{H}$ do problema de otimização quadrática é $2l \times 2l$. Os multiplicadores de Lagrange α_i, α_i^* serão diferentes de zero para os padrões do treinamento que estejam acima ou que estejam abaixo do tubo ε . Um padrão do treinamento não pode estar em ambos os lados do tubo (abaixo e acima do tubo), ou seja, ou α_i ou α_i^* será igual a zero. Para todos os padrões que estão dentro do tubo, ambos os multiplicadores serão iguais a zero. Desta forma, a multiplicação dos multiplicadores de Lagrange para um padrão $\mathbf{x}_i$ deve ser igual a zero, $\alpha_i \alpha_i^* = 0$ (HUANG; KECMAN; KOPRIVA, 2006).

Da mesma forma que ocorre com SVCs para utilizar o truque de *kernel*, basta substituir o termo $\mathbf{x}_i^T \mathbf{x}_j$ das equações (3.36) e (3.39) por uma função de kernel $K(\mathbf{x}_i, \mathbf{x}_j)$.

3.4 Métodos de treinamento

Os métodos clássicos matemáticos para resolução do problema de programação quadrática inerente a SVR são aplicados com sucesso em problemas com conjunto de treinamento reduzido. Porém a medida que a quantidade de padrões cresce há necessidade de mais capacidade computacional. Por outro lado, há um conjunto de algoritmos que utilizam a decomposição do conjunto de dados, a fim de reduzir a quantidade de memória necessária. Tais abordagens separam o conjunto em dois conjuntos disjuntos: o conjunto de trabalho com os multiplicadores que serão otimizados e os demais multiplicadores que serão mantidos fixos. O método de decomposição modifica somente os elementos que estão no conjunto de trabalho em cada iteração. Portanto, somente algumas colunas da matriz $\mathbf{Q}$ simétrica $\mathbf{Q} \in \mathbb{R}^{l \times l}$ são necessárias em cada iteração, o que reduz a quantidade de memória necessária para o armazenamento em cada iteração. A razão da redução de memória, neste contexto, é devido a solução ser obtida iterativamente ao invés de diretamente como na maioria dos métodos clássicos de Programação Quadrática (QP) (HUANG; KECMAN; KOPRIVA, 2006).

Dentre as alternativas para os métodos de QP pode-se citar o algoritmo de dados simples iterativo (*Iterative Single Data Algorithm*, ISDA) e o algoritmo de otimização sequencial mínima (*Sequential Minimal Optimization*, SMO). A característica mais importante na solução baseada no ISDA é que ele é aplicado sobre um padrão por vez. Enquanto que, o SMO utiliza

uma decomposição que opera sobre um conjunto de dois multiplicadores de Lagrange por vez. Para o algoritmo SMO, a melhor solução para esse conjunto pode ser encontrada analiticamente sem utilizar métodos de QP (HUANG; KECCMAN; KOPRIVA, 2006).

3.4.1 Programação Quadrática

Tanto em classificação como em regressão, o problema de aprendizado se resume em resolver problemas de otimização quadrática sujeitos as restrições de caixas (*box-constraints*) e a restrição de igualdade quando o termo viés é utilizado. O treinamento de SVMs em geral funciona de forma adequada em conjuntos de dados que não são grandes. No entanto, quando o número de padrões é superior, o problema de otimização quadrática se torna extremamente difícil de resolver com métodos de QP clássicos. Em tal situação pode não ser possível armazenar a matriz de *kernel* na memória (KECCMAN; HUANG; VOGT, 2005).

A Programação Quadrática tem como objetivo determinar a solução de um grupo especial de problemas de otimização (maximização ou minimização) que envolvem uma função quadrática com múltiplas variáveis sujeitas à restrições (NOCEDAL; WRIGHT, 2006). Os problemas podem então ser representados genericamente da maneira que segue

$$\min_{\mathbf{x}} g(\mathbf{x}) = \frac{1}{2} \mathbf{x}^T \mathbf{G} \mathbf{x} + \mathbf{x}^T \mathbf{d} \quad (3.40)$$

sujeito a

$$a_i^T \mathbf{x} = b_i \quad i \in \mathbb{A}, \quad (3.41)$$

$$a_i^T \mathbf{x} \geq b_i \quad i \in \mathbb{B}, \quad (3.42)$$

em que $\mathbf{G}$ é uma matriz simétrica de tamanho $n \times n$, $\mathbb{A}$ e $\mathbb{B}$ são conjuntos finitos de índices, e $\mathbf{d}$, $\mathbf{x}$ e $\{a_i\}$, $i \in \mathbb{A} \cup \mathbb{B}$, são vetores em $\mathbb{R}^n$ com n elementos cada. Neste contexto, para SVRs, o problema de programação quadrática tem a forma que segue

$$\min_{\boldsymbol{\alpha}} L_d(\boldsymbol{\alpha}) = \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H} \boldsymbol{\alpha} - \mathbf{p} \boldsymbol{\alpha}, \quad (3.43)$$

sujeito a

$$\mathbf{y}^T \boldsymbol{\alpha} = 0, \quad (3.44)$$

$$0 \leq \alpha_i \leq C, \quad i = 1, \dots, l,$$

em que, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_l, \alpha_1^*, \alpha_2^*, \dots, \alpha_l^*]$, $\mathbf{H} = [\mathbf{Q} \quad -\mathbf{Q}; -\mathbf{Q} \quad \mathbf{Q}]$ é a matriz Hessiana $2l \times 2l$, $\mathbf{Q}$ é a matriz de *kernel* $l \times l$, e $\mathbf{p} = [\varepsilon - y_1, \varepsilon - y_2, \varepsilon - y_l, \dots, \varepsilon + y_1, \varepsilon + y_2, \varepsilon + y_l]$ (KECCMAN;

HUANG; VOGT, 2005).

A partir dos problemas inerentes à QP anteriormente mencionados, houve a necessidade de desenvolver algoritmos e métodos de aprendizado iterativos que não precisem calcular a matriz de *kernel* por completa. Abordagens que fazem uso da decomposição do problema de otimização quadrática surgiram como alternativas (OSUNA; FREUND; GIROSI, 1997; VAPNIK, 1995).

3.4.2 *Sequential Minimal Optimization*

Sequential Minimal Optimization (SMO) foi inicialmente proposto em Platt (1999) e se trata de um método de treinamento que utiliza a abordagem de decomposição do conjunto. Na proposta, o algoritmo iterativamente seleciona subconjuntos com 2 multiplicadores de Lagrange e otimiza-os em relação a função. Tal abordagem melhora a convergência em termos de velocidade e seu ponto chave é que, ao se trabalhar com conjunto de 2 padrões, o subproblema de otimização pode ser solucionado analiticamente sem explicitamente utilizar um otimizador quadrático. A seleção deste conjunto de trabalho é um importante passo nos métodos de decomposição.

Dentre as formas de seleção do conjunto de trabalho há a seleção pelo par que mais viole as condições de otimalidade *Karush-Kuhn-Tucker* (KKT) e a seleção dada por informação de segunda ordem. Para este trabalho, o método de treinamento SMO para regressão utiliza informação de segunda ordem na seleção do conjunto de trabalho, a qual garante alcançar a convergência mais rapidamente (FAN; CHEN; LIN, 2005; CHEN; FAN; LIN, 2006).

Para estender SMO e aplicá-lo em problemas de regressão, as modificações consistem em uma alteração no sistema de equações para resolver o problema de otimização analiticamente dado que se trata do dobro do número de multiplicadores de Lagrange. A principal diferença de implementação do SMO está relacionado a quais critérios são utilizados para selecionar um novo conjunto de variáveis. O conjunto de 2 multiplicadores é escolhido utilizando informações de segunda ordem. A extensão do SMO (FAN; CHEN; LIN, 2005; CHEN; FAN; LIN, 2006) para SVR é dada sobre o problema de otimização quadrática da Equação (3.36), na notação que segue

$$\min_{\alpha} L_d(\alpha) = \frac{1}{2} \alpha^T \mathbf{H} \alpha - \mathbf{p} \alpha, \quad (3.45)$$

sujeito a

$$\begin{aligned} \mathbf{y}^T \boldsymbol{\alpha} &= 0, \\ 0 \leq \alpha_i \leq C, \quad i = 1, \dots, l, \end{aligned} \quad (3.46)$$

na qual, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_l, \alpha_1^*, \alpha_2^*, \dots, \alpha_l^*]$, $\mathbf{H} = [\mathbf{Q} \quad -\mathbf{Q} ; -\mathbf{Q} \quad \mathbf{Q}]$ é a matriz Hessiana $2l \times 2l$, $\mathbf{Q}$ é a matriz de *kernel* $l \times l$, e $\mathbf{p} = [\varepsilon - y_1, \varepsilon - y_2, \varepsilon - y_l, \dots, \varepsilon + y_1, \varepsilon + y_2, \varepsilon + y_l]$ (KECMAN; HUANG; VOGT, 2005). Ainda podendo ser reescrita em detalhes como segue

$$\begin{aligned} \min_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} f(\boldsymbol{\alpha}, \boldsymbol{\alpha}^*) &= \frac{1}{2} [\boldsymbol{\alpha}^T, \boldsymbol{\alpha}^{*T}] \begin{bmatrix} \mathbf{Q} & -\mathbf{Q} \\ -\mathbf{Q} & \mathbf{Q} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\alpha}^* \end{bmatrix} + [\varepsilon + \mathbf{y}^T, \varepsilon - \mathbf{y}^T] \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\alpha}^* \end{bmatrix} \\ \text{sujeito a } \mathbf{y}^T \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\alpha}^* \end{bmatrix} &= 0, \quad 0 \leq \alpha_i, \alpha_i^* \leq C, i = 1, \dots, l. \end{aligned} \quad (3.47)$$

Observe que a matriz $\mathbf{H}$ pode despendar alto custo em termos de memória para ser armazenada, dependendo do conjunto de treinamento, visto isso, os métodos de decomposição foram desenvolvidos para lidar com este problema.

O método de seleção do conjunto de trabalho com informação de segunda ordem pode ser diretamente aplicado para regressão. Mais detalhes sobre a forma como o conjunto de trabalho é escolhido pode ser encontrado em Fan, Chen e Lin (2005). Os multiplicadores de Lagrange são obtidos pelos sub-problemas que seguem

$$\begin{aligned} \min_{\boldsymbol{\alpha}_B} \frac{1}{2} [\boldsymbol{\alpha}_B^T \quad (\boldsymbol{\alpha}_N^k)^T] \begin{bmatrix} \mathbf{Q}_{BB} & -\mathbf{Q}_{BN} \\ -\mathbf{Q}_{NB} & \mathbf{Q}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix} + [\varepsilon + \mathbf{y}_B^T \quad \varepsilon - \mathbf{y}_N^T] \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix} \\ \text{sujeito a } \mathbf{y}_B^T \boldsymbol{\alpha}_B = -\mathbf{y}_N^T \boldsymbol{\alpha}_N^k, \quad 0 \leq \boldsymbol{\alpha}_B \leq C. \end{aligned} \quad (3.48)$$

Se o valor de $\alpha_{i,j} = K_{ii} + K_{jj} - 2K_{ij} \leq 0$ os multiplicadores serão obtidos da seguinte maneira

$$\begin{aligned} \min_{\boldsymbol{\alpha}_B} \frac{1}{2} [\boldsymbol{\alpha}_B^T \quad (\boldsymbol{\alpha}_N^k)^T] \begin{bmatrix} \mathbf{Q}_{BB} & -\mathbf{Q}_{BN} \\ -\mathbf{Q}_{NB} & \mathbf{Q}_{NN} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix} + [\varepsilon + \mathbf{y}_B^T \quad \varepsilon - \mathbf{y}_N^T] \begin{bmatrix} \boldsymbol{\alpha}_B \\ \boldsymbol{\alpha}_N^k \end{bmatrix} + \\ \frac{\tau - a_{i,j}}{4} ((\alpha_i - \alpha_i^k)^2 + (\alpha_j - \alpha_j^k)^2) \\ \text{sujeito a } \mathbf{y}_B^T \boldsymbol{\alpha}_B = -\mathbf{y}_N^T \boldsymbol{\alpha}_N^k, \quad 0 \leq \boldsymbol{\alpha}_B \leq C, \end{aligned} \quad (3.49)$$

tal que se trata da mesma Equação (3.48) agora com a adição do termo $\frac{\tau - a_{i,j}}{4} ((\alpha_i - \alpha_i^k)^2 + (\alpha_j - \alpha_j^k)^2)$, em que τ é uma constante de valor pequeno.

O algoritmo de decomposição SMO, utilizando o método de seleção do conjunto de trabalho com informação de segunda ordem é apresentado no Algoritmo 1 (FAN; CHEN; LIN, 2005; CHEN; FAN; LIN, 2006):

Algoritmo 1: *Sequential Minimal Optimization***Entrada:** $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_l, y_l)\}$ conjunto de treinamento ε desvio permitido**Resultado:** α conjunto de multiplicadores de Lagrange**1** Encontre α^1 como a solução inicial factível. Defina $k = 1$;**2 enquanto** α^k não for um ponto estacionário da Equação (3.47)**3** Encontre um outro conjunto de trabalho $B = \{i, j\} \in \{1, \dots, l\}$ através do método de seleção com informação de segunda ordem (FAN; CHEN; LIN, 2005).**4** Defina N e os demais índices $\neq i, j$;**5** Seja $a_{ij} = K_{ii} + K_{jj} - 2K_{ij}$.**6** **se** $a_{ij} > 0$ **7** Resolva o sub-problema da Equação (3.48)**8** **senão****9** Resolver a mesma equação com um termo adicional como na Equação (3.49), em que τ é um número positivo pequeno.**10** **fim****11** Defina $\alpha_N^{k+1} = \alpha_N^k$ e $k \leftarrow k + 1$.**12 fim****3.4.3 Iterative single data algorithm**

Ambos os algoritmos ISDA e SMO foram desenvolvidos como alternativas para solucionar o problema de programação quadrática quando o treinamento de SVM é realizado em grandes conjuntos de dados. O *Iterative single data algorithm* (ISDA) atualiza um multiplicador de Lagrange em cada iteração, diferentemente do SMO que resolve um sub-conjunto de 2 multiplicadores de Lagrange. Dentre as abordagens do ISDA, primeiramente foi proposto sem o termo viés b . Logo a formulação dual do problema de otimização quadrática apresentado na Equação (3.36) sem o termo explícito viés é dado como segue

$$\max_{\alpha, \alpha^*} \left\{ -\frac{1}{2} \sum_{i,j=1}^l (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \mathbf{x}_i^T \mathbf{x}_j - \varepsilon \sum_{i=1}^l (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \right\}, \quad (3.50)$$

sujeito a

$$\alpha_i, \alpha_i^* \in [0, C],$$

em que a Equação (3.50) é solucionada sem levar em consideração a restrição da Equação (3.38), na qual $\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0$. Posteriormente foi proposto o ISDA com um termo viés b explícito, em que o mesmo foi incorporado através da adição de uma constante positiva i/k à matriz de *kernel*. Essa nova abordagem é mais rápida em termos de tempo de treinamento (KECMAN; HUANG; VOGT, 2005).

O método de seleção do conjunto de trabalho é dado sobre o multiplicador de Lagrange que mais viola as condições KKT dentre todos α_i, α_i^* . Mais detalhes sobre o método de seleção podem ser encontrado em (KECMAN; HUANG; VOGT, 2005). Antes de apresentar a versão do ISDA para regressão, será apresentada as notações para SVC, visto que a formulação de ISDA para regressão está sobre ISDA para SVC. Então, para SVC temos que a notação em matriz do problema de otimização é dado como segue

$$\begin{aligned} \max L_d(\boldsymbol{\alpha}) &= -\frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H} \boldsymbol{\alpha} + \mathbf{p}^T \boldsymbol{\alpha}, \\ \text{sujeito a } \quad \alpha_i &\in [0, C], \quad i = 1, \dots, k, \end{aligned} \quad (3.51)$$

em que $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \dots, \alpha_l, \alpha_1^*, \alpha_2^*, \dots, \alpha_l^*]$, $\mathbf{H} = [\mathbf{Q} \quad -\mathbf{Q} ; -\mathbf{Q} \quad \mathbf{Q}]$ é a matriz Hessiana $2l \times 2l$, $\mathbf{Q}$ é a matriz de *kernel* $l \times l$, e $k = 2l$ e, por fim, $\mathbf{p} = [\varepsilon - y_1, \varepsilon - y_2, \varepsilon - y_l, \dots, \varepsilon + y_1, \varepsilon + y_2, \varepsilon + y_l]$ (KECMAN; HUANG; VOGT, 2005). Nesta abordagem de ISDA, o termo viés não é utilizado e a restrição da Equação (3.38) foi removida.

A ideia de encolhimento (*shrinking*) é reduzir o tamanho do problema de otimização da Equação (3.51) gradualmente pela remoção dos vetores suporte de borda e os possíveis não-vetores suporte durante o processo de otimização. Porém, o conjunto correto desses vetores suporte não pode ser encontrado sem resolver a Equação (3.51). Para isto, heurísticas são utilizadas para encolher o problema de otimização. Para isto, os algoritmos tentam adivinhar quais variáveis estão mais propensas a não serem vetores suporte ou serem vetores suporte de borda na solução final da Equação (3.51) através do histórico dessas variáveis nas últimas h iterações do algoritmo. Como resultado do encolhimento, o algoritmo precisa resolver o problema de otimização reduzido através de um laço interno como segue

$$\begin{aligned} \max L_d(\boldsymbol{\alpha}_A) &= -\frac{1}{2} \boldsymbol{\alpha}_A^T \mathbf{Q}_{AA} \boldsymbol{\alpha}_A + \mathbf{f}_A^T \boldsymbol{\alpha}_A, \\ \text{sujeito a } \quad (\boldsymbol{\alpha}_A)_t &\in [0, C], \quad t = 1, \dots, q, \end{aligned} \quad (3.52)$$

em que $\mathbf{A}$ de tamanho q corresponde aos índices restantes $\boldsymbol{\alpha}^{(*)}$ (ambos α_i e α_i^*) que não foram removidos pela heurística de encolhimento. Isso significa que o método realiza a otimização sobre $\boldsymbol{\alpha}_A$ que corresponde a aproximadamente 10% a 20% de todos os dados. Consequentemente, o problema de otimização será dado em $0,2l \times 0,2l$ ao invés de $l \times l$. A

diferença da Equação (3.52) quando aplicada para regressão é que usa-se $2q$ ao invés de q , ou seja, $t = 1, \dots, 2q$, visto que, ambos α_i e α_i^* estão inclusos em α_A .

O conjunto de trabalho está relacionado a $\alpha^{(*)}$ que será modificado ao passo que os demais são mantidos fixos. Os passos do método de treinamento ISDA para regressão são apresentados no Algoritmo 2. No algoritmo apresentado, considere um problema de regressão com uma matriz $\mathbf{X}_{treino} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_l]^T$ em que $\mathbf{X}_{treino} \in \mathbb{R}^{l \times m}$, em que m é o número de características, l o número de padrões, as saídas são $\mathbf{y} = [y_1, y_2, \dots, y_l]^T$. Seja I os índices correspondentes a todos os dados de treinamento, e o conjunto A ($A \subset I$) contendo os índices correspondentes a $\alpha^{(*)}$ que estão no problema de otimização reduzido apresentado na Equação (3.52). A^C é o complemento de A e I (HUANG; KECMAN; KOPRIVA, 2006).

Algoritmo 2: *Iterative single data algorithm***Entrada:** $h, C, \varepsilon, \mathbf{X}_{treino} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_l, y_l)\}$ **Resultado:** α_{oA} conjunto de multiplicadores de Lagrange

- 1 Seja $A = I = [1 : l]$.
- 2 Encontre z , o índice de A que mais viola as condições KKT (KECMAN; HUANG; VOGT, 2005).
- 3 **enquanto** não for a repetição h
 - 4 **se** o valor das condições KKT para $\mathbf{x}_z$ for menor que τ
 - 5 O sub-problema da Equação (3.52) está solucionado. Pule para a linha 22.
 - 6 **senão**
 - 7 **se** $\mathbf{x}_z$ está sobre o tubo ε
 - 8 $\alpha_z \leftarrow \alpha_z - \alpha_z^* - \frac{KKT}{K(\mathbf{x}_z, \mathbf{x}_z)}$
 - 9 **senão**
 - 10 $\alpha_z^* \leftarrow \alpha_z^* - \alpha_z - \frac{KKT}{K(\mathbf{x}_z, \mathbf{x}_z)}$
 - 11 **fim**
 - 12 $\alpha_z^{(*)} \leftarrow \min(\max(0, \alpha_z^{(*)} + \Delta \alpha_z^{(*)}), C)$
 - 13 **fim**
 - 14 Atualize as condições KKT para todos os índices i no conjunto A .
- 15 **fim**
- 16 Remova do conjunto A todos os índices que não violam as condições KKT e que α_i e α_i^* sejam ambos iguais a zero no conjunto. Isto é, o novo conjunto A agora é formado por todos que violam as condições KKT e que sejam vetores suporte.
- 17 **se** encontre $\alpha_z^{(*)}$ que mais viola as restrições KKT em A e se a violação for $> \tau$
 - 18 Volte para a linha 3.
- 19 **senão**
 - 20 Encontrar o novo A^C que é o complemento de A e I .
- 21 **fim**
- 22 Recalcule os valores das condições KKT para todos os índices i que estão em A^C
- 23 Encontre $\alpha_z^{(*)}$ que mais viola as restrições KKT em A^C .
- 24 **se** a violação for $> \tau$ ou A^C está vazio
 - 25 Pare e retorne a solução α_{oA} .
- 26 **senão**
 - 27 Volte para a linha 16.
- 28 **fim**

3.4.4 ε -evolutionary support vector regression

O ε -evolutionary support vector regression (ε -ESVR), uma abordagem que aplica algoritmos genéticos para a resolução do problema de otimização quadrática inerente a SVR linear foi proposta em (STOEAN *et al.*, 2006). O problema de otimização quadrática foi solucionado em sua formulação primal apresentada na Equação (3.29). No tocante a modelagem do problema para o arcabouço de algoritmos genéticos, o autor apresentou duas modelagens. Na primeira modelagem, o indivíduo incorpora $\mathbf{w}$, o viés b e as variáveis de folga $\{\xi_i\}_{i=1}^l$ e $\{\xi_i^*\}_{i=1}^l$. Na segunda, foi modelado incorporando $\mathbf{w}$ e o viés b . A função de aptidão foi dada pela minimização da equação do problema primal na primeira modelagem, e, para a segunda modelagem, foi acrescido à função o valor computado para as variáveis de folga como uma forma de penalizá-la. A função de aptidão para a segunda modelagem lida com a minimização da equação assim como o somatório das variáveis de folga.

A abordagem se mostra promissora quanto a sua modelagem, porém a metodologia dos experimentos realizados foi bastante modesta. O autor se restringiu à experimentos realizados sobre apenas uma base de dados. Além disso, por se tratar de uma abordagem alternativa para a resolução do problema de otimização quadrática inerente a SVR, carece de comparativos com os demais métodos de treinamentos estado-da-arte apresentados neste capítulo. O autor apresentou o resultado comparativo da sua proposta em relação ao SVR canônico e um modelo de regressão linear, porém não comparou seus resultados com métodos consagrados na literatura, como SMO, QP e ISDA.

3.5 Critérios de convergência

Cada um dos métodos de treinamento explanados anteriormente tem como critério de convergência as opções que seguem:

- *Feasibility gap*: o qual pode ser expresso como

$$\Delta = \frac{J(\beta) + L(\alpha)}{J(\beta) + 1}, \quad (3.53)$$

em que $J(\beta)$ é o objetivo primal e $L(\alpha)$ o dual. Depois de cada iteração, esta equação é computada. Se o valor for menor que uma tolerância especificada, então o algoritmo convergiu;

- Diferença de gradiente: o vetor de gradiente é computado, e se a diferença entre a iteração atual e a anterior for menor que um valor de tolerância especificado, o algoritmo convergiu;
- Maior violação KKT: onde depois de cada iteração, toda a violação KKT para todos os valores de α_i e α_i^* . Se a maior violação for menor que um valor de tolerância especificado, então o algoritmo convergiu.

3.6 Conclusão

Esse capítulo apresenta a teoria básica para a compreensão de Máquinas de Vetores Suporte para regressão. Primeiramente foi apresentado conceitos sobre SVM na sua formulação para classificação, e partir disto a extensão para regressão (SVR). O principal foco aqui é a apresentação dos aspectos inerentes a SVR: a programação quadrática presente nas formulações primal e dual desse regressor. Como visto, SVM se distingue de outros classificadores, como redes neurais artificiais, por prover um método que minimiza o risco empírico e o risco estrutural. O uso de variáveis de folga, garante robustez à conjuntos de dados ruidosos e o truque do kernel permite a aplicação do SVR em problemas não linearmente separáveis. Tais artifícios fazem com que esse método obtenha soluções melhores, no que diz respeito a generalização. Além disso, os principais métodos de treinamento foram apresentados. Métodos clássicos como QP, SMO e ISDA são detalhados. Uma solução alternativa que faz uso de algoritmos genéticos, também foi apresentada. De forma geral, o método mais utilizado atualmente no treinamento de SVR é o SMO, seguido por ISDA e QP. Embora os métodos clássicos obtenham boas soluções, ISDA não incorpora totalmente as restrições do SVR.

ALGORITMOS GENÉTICOS NO TREINAMENTO DE SVR

Neste capítulo é apresentada nossa proposta que se denomina Regressores de Vetores Suporte Evolucionários (ESVR). A nossa proposta se concentra em solucionar o problema de otimização quadrática de SVR em sua formulação dual através de algoritmos genéticos. Os conceitos e aspectos relacionados à AGs e SVRs foram apresentados no Capítulo 2 e Capítulo 3, respectivamente. Para uma modelagem bem sucedida de algoritmos genéticos, são necessários, dentre outros aspectos, uma boa representação genética e operadores de mutação e reprodução adequados, o mesmo pode ser dito para a função de aptidão. A seguir são apresentados os elementos que compõem o ESVR.

4.1 Visão geral

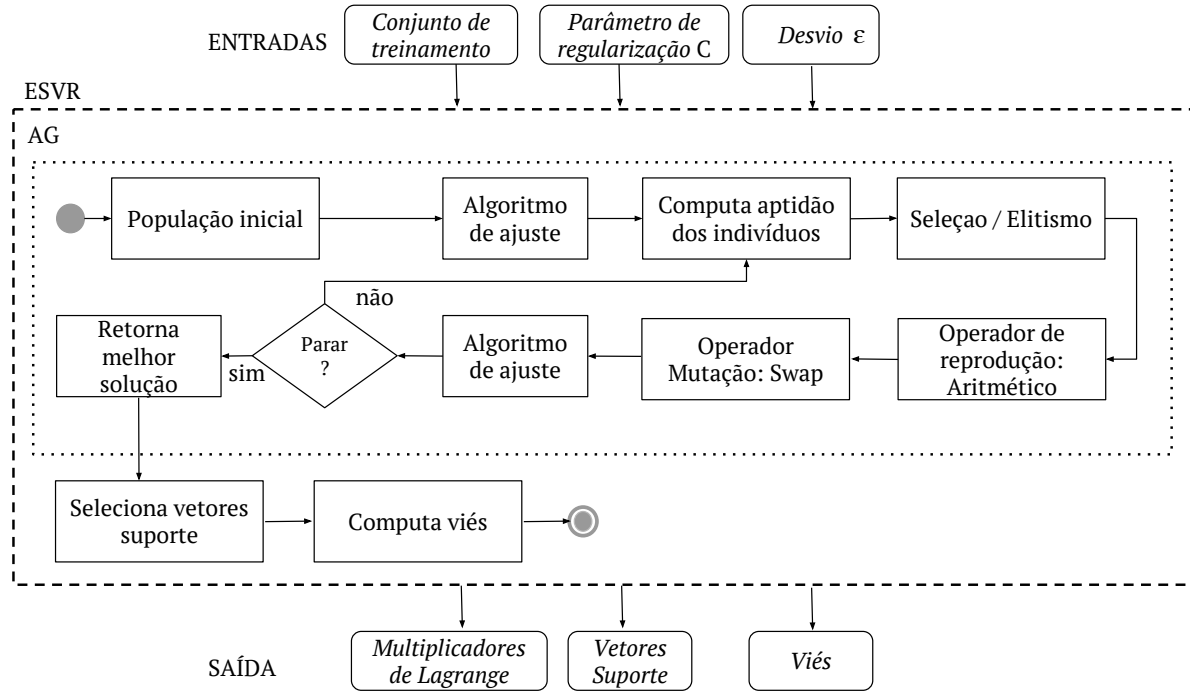
A proposta objetiva encontrar os multiplicadores de Lagrange através de AGs mono objetivo. ESVR tem como parâmetros de entrada: (i) o conjunto de treinamento, (ii) o parâmetro de regularização C e (iii) o desvio ε . Como saída: (i) o viés, (ii) os multiplicadores de Lagrange; e (iii) a relação de vetores suportes, os quais são a relação dos padrões do treinamento que estão associados com os multiplicadores de Lagrange diferentes de zero. Primeiramente, uma população inicial é gerada aleatoriamente, na qual, cada indivíduo representa uma possível solução, ou seja, um conjunto de multiplicadores de Lagrange. Em cada geração, até que ESVR alcance seu critério de parada, os seguintes passos são executados:

1. O elitismo é aplicado;
2. Uma fração de novos indivíduos são gerados através do operador de reprodução;
3. Os indivíduos restantes são gerados através do operador de mutação.

Finalmente, quando o critério de parada for alcançado, o qual é determinado por: (i) um certo limite de gerações for alcançada ou (ii) a média de mudanças relacionada ao melhor valor de aptidão for menor ou igual a um determinado valor; o ESVR finaliza e retorna os multiplicadores de Lagrange e o valor do

viés. Vale ressaltar que aqueles indivíduos que violam as restrições passam por um algoritmo de ajuste, o qual será apresentado na seção 4.3. O fluxo do ESVR é apresentado na Figura 11.

Figura 11: Fluxo do ESVR.



Fonte: autor.

Nas próximas seções são apresentados em detalhes os aspectos referentes a representação genética, operadores de mutação e reprodução, função de aptidão e o algoritmo de ajuste.

4.2 População e representação genética

A população consiste em um conjunto de soluções (indivíduos) que serão evoluídos durante o processo de AGs. A representação genética está relacionada a estrutura destes indivíduos. A solução para o problema de otimização quadrática de SVR é um conjunto de multiplicadores de Lagrange, dos quais, cada par α_i e α_i^* está associado a uma amostra existente no conjunto de treinamento. Cada multiplicador de Lagrange pode assumir valores entre 0 e C.

Desta forma, uma possível representação genética de uma solução pode ser dada por um vetor que contenha dois valores de multiplicadores de Lagrange para cada amostra do treinamento, ou seja, um vetor de tamanho $2l$. Porém, o espaço de busca seria maior visto que dentre as restrições inerentes a SVR temos que $\alpha_i \alpha_i^* = 0$. No processo de AGs, facilmente haveriam soluções que não seguiriam a esta restrição. Ao analisar o problema, tem-se que este conjunto pode ser reduzido pela metade. Como visto no Capítulo 3, $\alpha_i, \alpha_i^* \in [0, C]$ e $\alpha_i \alpha_i^* = 0$, ou seja, α_i ou α_i^* é igual a zero, pois não pode haver um conjunto de multiplicadores de Lagrange ambos diferentes de zero. Tendo isto em mente e observando a

Equação (3.36)

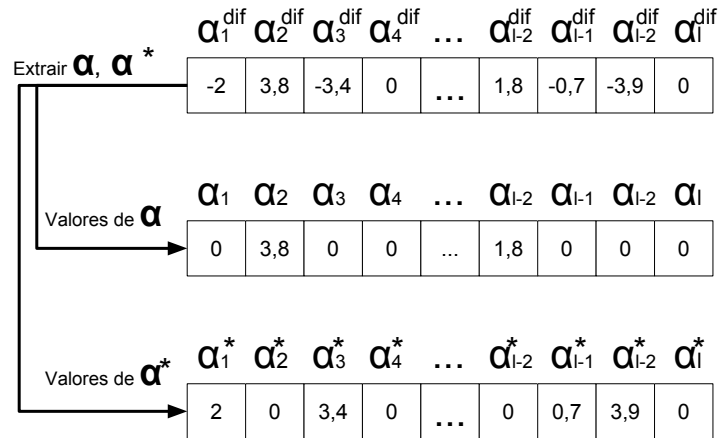
$$\max_{\alpha, \alpha^*} \left\{ -\frac{1}{2} \sum_{i,j=1}^l (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \mathbf{x}_i^T \mathbf{x}_j - \varepsilon \sum_{i=1}^l (\alpha_i + \alpha_i^*) + \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*) \right\},$$

considere então $\alpha_i^{dif} = (\alpha_i - \alpha_i^*)$. Pela equação e restrições de intervalo anteriormente mencionadas, o valor de $\alpha_i^{dif} \in [-C, C]$ e os valores de α_i e α_i^* podem ser recuperados como segue

$$\begin{aligned} \alpha_i, \alpha_i^* &= 0 & \text{se } \alpha_i^{dif} &= 0, \\ \alpha_i &= 0 \text{ e } \alpha_i^* = |\alpha_i^{dif}| & \text{se } \alpha_i^{dif} < 0, \\ \alpha_i &= |\alpha_i^{dif}| \text{ e } \alpha_i^* = 0 & \text{se } \alpha_i^{dif} > 0. \end{aligned} \quad (4.1)$$

Desta forma, a representação genética é dada por um indivíduo de l genes $\boldsymbol{\alpha}^{dif} = [(\alpha_1 - \alpha_1^*), (\alpha_2 - \alpha_2^*), \dots, (\alpha_i - \alpha_i^*), \dots, (\alpha_l - \alpha_l^*)]$, $\alpha_i^{dif} \in [-C, C]$, onde cada gene α_i^{dif} está relacionado a uma amostra do treinamento, do qual podem ser extraídos os dois multiplicadores de Lagrange. Um exemplo da extração dos multiplicadores de Lagrange a partir de um indivíduo qualquer é apresentado na Figura 12, na qual temos que os valores atribuídos a α_k^{dif} , $k = 1, \dots, l$ estão entre -4 e 4 , para $C = 4$. Iniciando pelo primeiro gene, temos que valor de $\alpha_1^{dif} = -2$ é negativo. Ao aplicar a Equação (4.1), temos que $\alpha_1 = 0$ e $\alpha_1^* = 2$ visto que $|-2| = 2$. O gene 4 (α_4^{dif}) é igual zero, para essa situação ambos os multiplicadores de Lagrange são iguais a zero. De forma mais direta, quando o gene i for negativo, significa que o $\alpha_i^* > 0$ e quando o gene for positivo, o $\alpha_i > 0$, visto que apenas um dos multiplicadores pode ser diferente de zero.

Figura 12: Representação genética, considerando o parâmetro $C = 4$.



Fonte: autor.

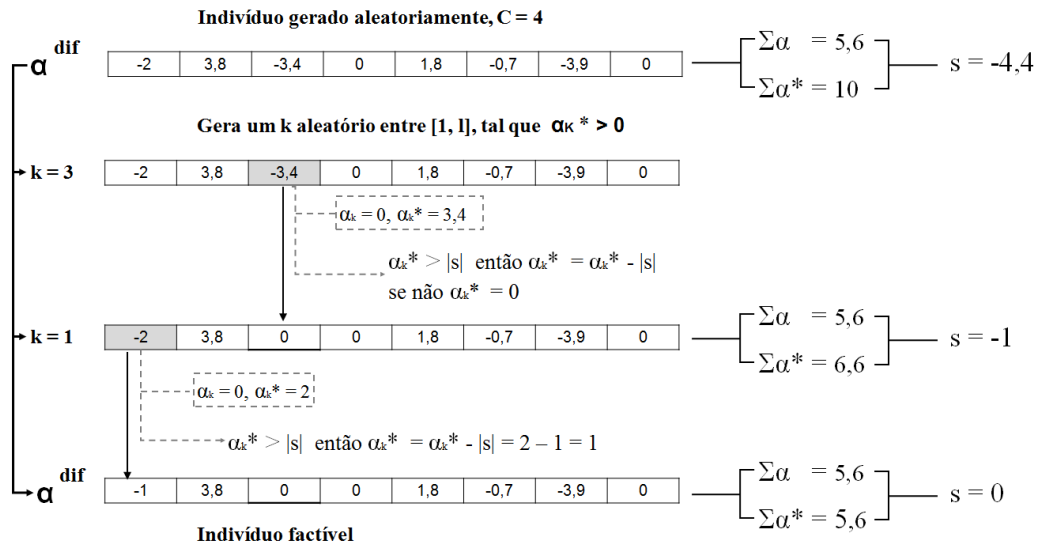
Os indivíduos nesta representação não violam a restrição apresentada na Equação (3.38) ($\alpha_i \alpha_i^* = 0$ e $\alpha_i, \alpha_i^* \in [0, C]$). No entanto, violam a segunda restrição dada por $\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0$ que pode ser vista na Equação (3.37). Tal violação pode ser contornada por um algoritmo de ajuste, o qual será explicado na Seção 4.3.

Do exposto, a iniciação da população é realizada de forma aleatória com distribuição uniforme com valores no intervalo $[-C, C]$.

4.3 Algoritmo de ajuste

A nossa representação genética atende a restrição apresentada na Equação (3.37), pois na configuração do algoritmo genético foi definido o limite de valores atribuídos aos genes de acordo com o valor do parâmetro C do SVR. Por outro lado, estes indivíduos podem violar a restrição apresentada na Equação (3.38). Geralmente, quando um indivíduo viola as restrições do problema existem três estratégias comuns na literatura (MICHALEWICZ; JANIOW, 1991): o ajuste, penalizá-lo pela violação ou removê-lo da população. Na proposta, optou-se por tornar os indivíduos soluções factíveis através de um algoritmo de ajuste. O algoritmo visa corrigir um indivíduo para que a restrição apresentada na Equação (3.38), $\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0$, seja satisfeita. A ideia principal do algoritmo de ajuste é iterativamente reduzir uma quantidade de $\sum_{i=1}^l \alpha_i$ ou $\sum_{i=1}^l \alpha_i^*$ até que α^{dif} esteja de acordo com a restrição. Na Figura 13 é apresentado o processo de ajuste aplicado ao indivíduo apresentado na Figura 12. A restrição de que $\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0$ significa que $\sum_{i=1}^l \alpha_i = \sum_{i=1}^l \alpha_i^*$.

Figura 13: Aplicação do algoritmo de ajuste. Considere o parâmetro $C = 4$.



Fonte: autor.

Neste raciocínio, é verificado se $\sum_{i=1}^l \alpha_i > \sum_{i=1}^l \alpha_i^*$ então uma quantidade $s = \sum_{i=1}^l \alpha_i - \sum_{i=1}^l \alpha_i^*$ será deduzida de $\sum_{i=1}^l \alpha_i$, se não, uma quantidade s será deduzida de $\sum_{i=1}^l \alpha_i^*$ até que a restrição seja satisfeita. Se não for possível deduzir s caso $s > \alpha_i^{(*)}$, então duas alternativas são escolhidas aleatoriamente, ou o multiplicador de Lagrange que está sendo atualizado é zerado, desta forma há uma redução na quantidade de vetores suporte, visto que só pode haver um multiplicador de Lagrange não zero $\alpha_i^{(*)}$. Ou, é atribuído ao outro multiplicador o valor de C , ou seja, supondo que o multiplicador que está sendo atualizado é α_i , ocorre então que ou $\alpha_i = 0$ ou $\alpha_i^* = C$. O processo de ajuste é apresentado no Algoritmo 3.

Algoritmo 3: Algoritmo de ajuste de indivíduos**Entrada:** α^{dif} : indivíduo a ser ajustado C : parâmetro que limita os valores de $\alpha^{(*)}$ **Resultado:** α^{dif} indivíduo ajustado

```

1   $\alpha, \alpha^* \leftarrow$  recupera os valores de  $\alpha$  e  $\alpha^*$  a partir de  $\alpha^{dif}$  através da Equação (4.1)
2   $s \leftarrow \sum_{i=1}^l (\alpha_i - \alpha_i^*)$  computa a soma dos multiplicadores de Lagrange
3  enquanto  $|s| \neq 0$                                 //O algoritmo para quando  $\sum_{i=1}^l \alpha_i = \sum_{i=1}^l \alpha_i^*$ 
4      se  $s > 0$                                         //significa que precisa ser deduzido  $s$  de  $\alpha$ 
5           $k \leftarrow \text{RAND}[1, l]$                     //tal que  $\alpha_k > 0$ , para manter os multiplicadores que estão zerados
6          se  $\alpha_k > |s|$ 
7               $\alpha_k \leftarrow \alpha_k - |s|$             //deduz a soma  $s$  de  $\alpha_k$ 
8          senão
9               $\alpha_k \leftarrow 0$                         // $\alpha_k$  é maior que a diferença. Ou zera este multiplicador, o que
                                                         significa que  $\mathbf{x}_k$  deixa de ser vetor suporte
10             se  $\text{RAND}[0, 1] < 0.5$ 
11                  $\alpha_k^* \leftarrow C$                 //ou atribui o valor de  $C$  ao  $\alpha_k^*$  (que era anteriormente zero)
12             senão
13                 fim
14         fim
15          $\alpha_k^{dif} \leftarrow (\alpha_k - \alpha_k^*)$     //atualiza o gene do indivíduo
16     senão
17          $k \leftarrow \text{RAND}[1, l]$                     //tal que  $\alpha_k^* > 0$ , para manter os multiplicadores que estão zerados
18         se  $\alpha_k^* > |s|$ 
19              $\alpha_k^* \leftarrow \alpha_k^* - |s|$         //deduz a soma  $s$  de  $\alpha_k^*$ 
20         senão
21              $\alpha_k^* \leftarrow 0$                         // $\alpha_k^*$  é maior que a diferença. Ou zera este multiplicador, o que
                                                         significa que  $\mathbf{x}_k$  deixa de ser vetor suporte
22             se  $\text{RAND}[0, 1] < 0.5$ 
23                  $\alpha_k \leftarrow C$                 //ou atribui o valor de  $C$  ao  $\alpha_k$  (que era anteriormente zero)
24             senão
25                 fim
26         fim
27          $\alpha_k^{dif} \leftarrow (\alpha_k - \alpha_k^*)$     //atualiza o gene do indivíduo
28     fim
29      $s \leftarrow \sum_{i=1}^l (\alpha_i - \alpha_i^*)$         //atualiza a soma  $s$  após a alteração do valor de  $\alpha_k^{dif}$ 
30 fim
31 Retorne  $\alpha^{dif}$ 

```

4.4 Função de aptidão

A função de aptidão é responsável por guiar o algoritmo genético no espaço de busca do problema. A mesma visa avaliar o quão boa é uma solução (indivíduo), ou seja, avaliar quão bom é um conjunto de multiplicadores de Lagrange. Para encontrar os multiplicadores de Lagrange, a Equação (3.36) deve ser maximizada. Então, a função de aptidão da nossa proposta é dada em função dessa equação. O valor de aptidão de um indivíduo $F(\boldsymbol{\alpha}^{dif})$ em termos de minimização pode ser calculado por

$$F(\boldsymbol{\alpha}^{dif}) = \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \mathbf{x}_i^T \mathbf{x}_j + \varepsilon \sum_{i=1}^l (\alpha_i + \alpha_i^*) - \sum_{i=1}^l y_i (\alpha_i - \alpha_i^*), \quad (4.2)$$

em que α_i e α_i^* são os multiplicadores de Lagrange que são obtidos a partir de $\boldsymbol{\alpha}^{dif}$ segundo a Equação (4.1).

4.5 Operadores genéticos

Como dito, os operadores genéticos são responsáveis por gerar novas soluções, são eles o operador de reprodução e o de mutação. O operador de reprodução é capaz de gerar novos indivíduos por meio da troca de material genético dos pais, enquanto o operador de mutação modifica um indivíduo a fim de inserir diversidade na população e explorar, desta forma, outros espaços de busca.

Dentre os operadores de reprodução, foi utilizado o operador aritmético de recombinação intermediária (*Intermediate Recombination*) (MÜHLENBEIN; SCHLIERKAMP-VOOSEN, 1993). Neste operador, os indivíduos são gerados com valores de genes próximos aos valores dos genes dos pais. Seja $\mathbf{p} = [p_1, \dots, p_n]$ e $\mathbf{q} = [q_1, \dots, q_n]$ os pais de um indivíduo $\mathbf{z}$. O i -ésimo gene do indivíduo $\mathbf{z}$ é obtido por

$$z_i = \gamma_i p_i + (1 - \gamma_i) q_i, \quad i = 1, \dots, n, \quad (4.3)$$

em que γ_i é um fator de escala escolhido aleatoriamente no intervalo $[-g, 1 + g]$. Nesse operador, g define o intervalo de valores no qual os valores para o indivíduo são gerados. Quando $g = 0$, o intervalo de $\mathbf{z}$ é o mesmo dos pais $\mathbf{p}$ e $\mathbf{q}$. O valor de g foi mantido igual a zero para evitar que fosse gerado um indivíduo cujo gene fosse maior que C . Desta forma, o indivíduo gerado não viola a restrição $\mathbf{z} \in [-C, C]$ apresentada na Equação (3.37). Com relação ao operador de mutação, o utilizado foi operador de permutação (DAVIS, 1991). O mesmo seleciona dois genes e permuta seus valores.

Em ambos os operadores de reprodução e mutação, $\mathbf{z}$ viola a restrição $\sum_{i=1}^l (\alpha_i - \alpha_i^*) = 0$ apresentada na Equação (3.38). Esses indivíduos devem passar pelo algoritmo de ajuste, para torná-los soluções factíveis.

4.6 Elitismo

No fluxograma mais comum em algoritmos genéticos, os indivíduos selecionados para participar do processo de reprodução e mutação são, geralmente, substituídos pelos novos indivíduos gerados. Entenda que, estes progenitores são, na maioria das vezes, indivíduos com alto valor de aptidão. Por mais que os novos indivíduos possam representar melhores soluções, seria interessante manter as melhores soluções da geração atual na próxima população. Para realizar isto, há um mecanismo, chamado de elitismo, que mantém uma determinada fração da população da geração atual na seguinte. Este mecanismo foi adotado na nossa proposta, onde uma fração dos melhores indivíduos da população atual são adicionados à população da geração seguinte.

4.7 Cômputo do termo viés

O termo viés b é computado ao término do processo do ESVR. A partir do momento que os multiplicadores de Lagrange foram encontrados, o valor do viés b é obtido pela Equação (3.39), quando reescrita da seguinte maneira

$$b = \frac{1}{l} \sum_{i=1}^l \left(y_i - \sum_{i=1}^l (\alpha_i - \alpha_i^*) \mathbf{x}_i^T \mathbf{x} \right), \quad (4.4)$$

em que no final do treinamento os valores dos multiplicadores de Lagrange foram encontrados, e por se tratar de método de aprendizado supervisionado, a informação da variável correspondente y_i no conjunto de treinamento é conhecida.

4.8 Conclusão

Este capítulo apresenta um método de treinamento de SVR denominado ESVR, sendo apresentados os aspectos relacionados a modelagem e adequação do problema de otimização para o arcabouço de algoritmos genéticos. Dentre outros aspectos, podemos citar: a representação genética, operadores de reprodução e mutação, a função de aptidão, algoritmo de ajuste e critério de parada. Embora os métodos clássicos obtenham boas soluções, as restrições da Equação (3.37) e Equação (3.38) não são totalmente satisfeitas, como é o caso do ISDA, que omite o termo viés desconsiderando a Equação (3.38) como explanado no Capítulo 3. A proposta incorpora totalmente as restrições do SVR de forma a obter soluções bastante similares ao que se encontra com o uso do QP, SMO e ISDA, com a vantagem de serem soluções totalmente válidas, no que diz respeito à adequação do método de treinamento ao problema de otimização do SVR.

SIMULAÇÕES COMPUTACIONAIS

Neste capítulo é apresentada a metodologia utilizada nas etapas de treinamento, otimização de hiperparâmetros e teste para os métodos QP, SMO, ISDA e para a nossa proposta ESVR. Em seguida são apresentadas as características dos conjuntos de dados utilizadas para validação da proposta. Também são apresentados resultados das simulações computacionais realizadas com os métodos de treinamento de SVR. Para finalizar, é realizada uma análise sobre os resultados, bem como uma comparação entre a proposta e os métodos de treinamento supracitados.

5.1 Metodologia

Para avaliar nossa proposta foram realizados experimentos com 8 conjuntos de dados reais, das quais 7 estão disponíveis no repositório de aprendizado de máquinas da UCI (BACHE; LICHMAN, 2013). Os conjuntos de dados utilizados neste trabalho foram: (i) *Auto price* (AUT), (ii) *Servo* (SER), (iii) *Breast cancer wisconsin* (BCR), (iv) *CPU*, (v) *Boston*, (vi) *Ailerons*, e (vii) *Elevators* e a (viii) *Motorcycle* descrita em Eubank (1999). A descrição, abreviação, quantidade de atributos e quantidade de padrões destes conjuntos de dados são apresentados na Tabela 1. A proposta foi submetida aos conjuntos de dados que contém um número reduzido de padrões, como também a conjuntos com mais de 9500 padrões. Estes experimentos se justificam, em razão dos conjuntos possuírem, geralmente, um alto volume de dados.

Tabela 1: Detalhes dos conjuntos de dados das simulações

Conjunto de dados	Abreviação	#Variáveis	#Amostras
Auto price	AUT	15	159
Servo	SER	4	167
Breast Cancer Wisconsin	BCR	32	194
CPU	CPU	6	209
Boston	BOS	13	506
Motorcycle	MOT	1	133
Ailerons	AIL	5	7129
Elevators	ELE	6	9517

Fonte: autor.

Independente do conjunto de dados e do método de treinamento utilizado, todos os padrões são submetidos primeiramente a um pré-processamento, chamado de normalização. A normalização ajusta os valores dos atributos de diferentes conjuntos de dados, que naturalmente estão em escalas diferentes, para uma mesma escala. Dentre os diversos tipos de normalização, tais como normalização min-max, normalização z-score (média zero) e normalização por escalonamento decimal (HAN; PEI; KAMBER, 2011), utilizou-se a normalização de média zero. Nesta normalização os valores a_i de um atributo A são normalizados para a'_i com base na sua média e em seu desvio padrão, como segue

$$a'_i = \frac{a_i - \eta_A}{\sigma_A} \quad , \quad (5.1)$$

em que η_A e σ_A são a média e desvio padrão da variável A , respectivamente. Com a aplicação dessa normalização, os padrões estarão dispostos sob uma média 0 e desvio padrão 1.

Há um conjunto de parâmetros, chamado de hiperparâmetros, que independem do método utilizado para treinamento de SVR, de tal maneira que estão relacionados ao problema de otimização e não em como resolvê-lo. Em SVRs, os hiperparâmetros são a constante C para regularização, o máximo desvio permitido ε e os parâmetros relacionados ao *kernel*. A escolha destes hiperparâmetros está diretamente relacionada ao desempenho do SVR. No entanto, não se sabe a priori qual é o melhor valor de C , ε e σ (*kernel* gaussiano) para determinado conjunto de dados e assim, faz-se necessário algum tipo de seleção de modelo para encontrar os valores de tais hiperparâmetros.

Existem diferentes técnicas com essa finalidade, a mais utilizada dentre elas e também aplicada neste trabalho é a busca em grade (*grid search*) combinada com a técnica de validação cruzada (*cross-validation*). A busca em grade é uma busca exaustiva sobre um conjunto de hiperparâmetros que se deseja otimizar. Cada combinação de valores é então testada para determinado conjunto de dados e seus respectivos valores de erro são armazenados. No final do processo, a combinação de hiperparâmetros de menor erro é escolhida. O tempo de execução desta busca depende da granularidade da grade de busca e do número de hiperparâmetros envolvidos. A outra técnica mencionada, a validação cruzada, tem como principal característica a divisão do conjunto de dados para a estimação do erro. Parte dos padrões são utilizados para treinamento e o restante deles para a validação e estimação do erro. A principal característica da validação cruzada é o fato dela utilizar todos os dados para teste pelo menos uma vez.

Existem três principais tipos de validação cruzada, a saber: *holdout*, *k-fold* e *leave-on-out*. No método *holdout*, o conjunto de dados é dividido em dois subconjuntos, um para treinamento e outro para teste. No método *k-fold* o conjunto de dados é dividido em k subconjuntos mutuamente exclusivos de mesmo tamanho e a partir disso, um subconjunto é utilizado para validação, enquanto os demais $k-1$ conjuntos são utilizados para treinamento e teste¹. Por fim, para o método *leave-one-out* tem-se um caso específico de *k-fold*, em que k é igual ao número total de instâncias l .

A combinação da busca em grade com a validação cruzada *holdout* e *k-fold* foi utilizada nos

¹Esse processo é realizado k vezes alternando o subconjunto de validação.

experimentos. No método *holdout*, a divisão adotada foi de 80% do conjunto de dados para treinamento e os 20% restantes para estimação das métricas de comparação, tais como, erro quadrático médio (MSE), tempo de treinamento e teste e o número de vetores suporte (#SV). O método *k-fold* para um $k = 5$ foi então utilizado sobre os 80% dos padrões que compreendem o conjunto de treinamento.

Os experimentos foram executados 30 vezes para cada par conjunto de dados/método de treinamento com o *kernel* linear e também com o *kernel* gaussiano. O hiperparâmetro ε foi fixado e o parâmetro do *kernel* gaussiano σ foi obtido por meio da busca em grade com validação cruzada, onde $\sigma \in [2^{-15}, 2^{-14}, \dots, 2^3]$. No tocante ao hiperparâmetro C , destaca-se que o mesmo foi obtido de forma analítica como descrito em Cherkassky e Ma (2004). Neste trabalho, o autor determina o valor de C a partir de valores das variáveis dependentes y do conjunto de treinamento. Além disso, como se sabe, este parâmetro define a faixa de valores dos multiplicadores de Lagrange. A forma analítica para a definição do hiperparâmetro C é a seguinte: $C = \max(|\bar{y} + 3\sigma_y|, |\bar{y} - 3\sigma_y|)$, em que $\bar{y}$ e σ são a média e desvio padrão de y no conjunto de treinamento, respectivamente. Mais detalhes podem ser encontrados em Cherkassky e Ma (2004).

Com relação aos conjuntos de dados *Ailerons* e *Elevators*, nos quais $l \in [7000, 10000]$, destaca-se que foi utilizado o método de agrupamento *k-medóide* (KAUFMAN; ROUSSEEUW, 1990) sobre o conjunto de treinamento com o objetivo de reduzir o conjunto de dados. Este método de agrupamento é similar ao *k-médias* (ARTHUR; VASSILVITSKII, 2007), sendo que o objetivo de ambos é dividir o conjunto de dados em k agrupamentos. No *k-médias* o centro do subconjunto é dado como a média do subconjunto, chamado de centróide. Já no *k-medóide*, o centro do subconjunto é dado por um membro do próprio subconjunto, chamado de medóide. O valor de k foi definido como 500, visto que valores superiores para o conjunto de treinamento exigem mais de algoritmos genéticos, no que se refere ao custo computacional.

Em cada uma dessas 30 realizações, os subconjuntos de treinamento e teste foram selecionados de forma aleatória, vale ressaltar que esse mesmo conjunto foi utilizado por todos os métodos de treinamento naquela execução. Além disso, tais conjuntos de treinamento e teste, de cada uma das 30 realizações, foram mantidos tanto para as simulações com o *kernel* linear quanto para o *kernel* gaussiano. Os resultados a serem apresentados são estimativas tomadas, tal como a média geral das 30 realizações em cada conjunto de dados por método de treinamento. Os parâmetros que são inerentes a cada método de treinamento foram mantidos como na implementação padrão do MATLAB, já os parâmetros das propostas foram selecionados por tentativa-e-erro. Os diversos valores de parâmetros são mostrados na Tabela 2.

Além disso, foi realizada uma verificação de equivalência estatística entre a proposta e os demais métodos de treinamento, através do teste de Friedman (DEMSAR, 2006). A hipótese nula indica que os métodos de treinamentos são equivalentes, ao passo que a hipótese alternativa indica o contrário. Se o teste fornecer um nível de significância menor ou igual ao valor limite definido como 0,01, significa que os dados observados são inconsistentes com a hipótese nula, rejeitando-a. O teste estatístico de

Tabela 2: Parâmetros dos métodos

Método	Parâmetro	Valor escolhido manualmente
ESVR	Tamanho da população	60
	Limite de gerações	3000
	Mutação	25%
	Reprodução	70%
	Elitismo	5%
	Limite para estagnação	$1.0e-4$
SMO	Máximo de iterações	15000
	Tolerância (λ)	0.001
QP	Máximo de iterações	15000
	Tolerância	0.001
ISDA	Máximo de iterações	15000

Fonte: autor.

Friedman, por ser não paramétrico, não depende de suposições sobre a distribuição dos dados.

5.2 Resultados e discussões

Os resultados dos experimentos são apresentados por meio de tabelas e gráficos de barras, gráficos de dispersão para cada uma das 30 realizações por conjunto de dados. Os resultados referentes a MSE, tempo de treino e tempo de teste, quando apresentados em gráficos, passaram por uma normalização para que fosse possível apresentá-los conjuntamente em um único gráfico, visto que esses valores diferem entre conjunto de dados e, se mantidos, inviabilizariam a apresentação conjunta. A normalização min-max foi realizada de tal maneira que os valores foram modificados para a escala $\in [0,1]$, e nesta normalização o valor 0 foi definido como o mínimo. Com relação ao número de vetores suporte, os mesmos também diferem entre conjuntos de dados e, semelhantemente, para uma melhor avaliação dos resultados, estes valores foram apresentados como uma razão entre a quantidade de vetores suporte utilizados e o tamanho do conjunto de treinamento: $\#sv/l$. Esta razão representa a porcentagem de vetores suporte em relação ao total de padrões do conjunto de treino.

As métricas de desempenho são apresentadas em termos da média e desvio padrão do MSE, média do tempo necessário para treinamento (TR) e teste (TE) e ainda em termos do número de vetores suportes ($\#SV$). Os resultados referentes à média do conjunto de teste sobre as 30 realizações para o SVRs com *kernel* linear treinados pelos métodos SMO, QP, ISDA e ESVR estão na Tabela 3, enquanto os resultados para *kernel* gaussiano são apresentados na Tabela 4. O método que obteve menor valor para MSE está destacado em negrito. Também são apresentados os resultados do teste estatístico de Friedman (TF), em que ✓ indica uma relação de equivalência dos outros métodos em relação ao ESVR, enquanto ✗ indica

uma diferença entre eles.

Tabela 3: Resultados da média das 30 realizações com *kernel* linear.

Base	Método	MSE	TF (MSE)	#SV	TR (s)	TE (s)
AUT	SMO	$7,90\text{e}+06 \pm 3,89\text{e}+06$	✗	$127,00 \pm 0,00$	4,43e-02	$5,43\text{e}-04$
	QP	$7,82\text{e}+06 \pm 3,81\text{e}+06$	✗	$127,00 \pm 0,00$	$2,62\text{e}-01$	$5,62\text{e}-04$
	ISDA	$7,80\text{e}+06 \pm 3,76\text{e}+06$	✗	$126,17 \pm 1,66$	$6,63\text{e}-02$	$5,78\text{e}-04$
	ESVR	$2,04\text{e}+07 \pm 1,41\text{e}+07$	-	$113,07 \pm 10,24$	$7,83\text{e}+00$	$1,85\text{e}-05$
CPU	SMO	$6,27\text{e}+03 \pm 6,54\text{e}+03$	✓	$167,00 \pm 0,00$	4,00e-02	$5,32\text{e}-04$
	QP	$6,24\text{e}+03 \pm 6,49\text{e}+03$	✓	$166,97 \pm 0,18$	$3,90\text{e}-01$	$5,32\text{e}-04$
	ISDA	$6,37\text{e}+03 \pm 6,70\text{e}+03$	✓	$150,43 \pm 5,91$	$6,74\text{e}-02$	$5,31\text{e}-04$
	ESVR	$5,21\text{e}+03 \pm 5,14\text{e}+03$	-	$132,77 \pm 21,85$	$8,92\text{e}+00$	$2,42\text{e}-05$
SER	SMO	$1,90\text{e}+00 \pm 7,68\text{e}-01$	✗	$133,90 \pm 0,31$	1,55e-02	$5,30\text{e}-04$
	QP	$1,90\text{e}+00 \pm 7,77\text{e}-01$	✗	$133,90 \pm 0,31$	$1,93\text{e}-01$	$5,22\text{e}-04$
	ISDA	$1,91\text{e}+00 \pm 7,67\text{e}-01$	✗	$133,00 \pm 0,91$	$1,96\text{e}-02$	$5,58\text{e}-04$
	ESVR	$1,27\text{e}+00 \pm 4,58\text{e}-01$	-	$102,83 \pm 18,01$	$6,70\text{e}+00$	$1,31\text{e}-05$
BCR	SMO	$1,17\text{e}+03 \pm 2,02\text{e}+02$	✗	$154,97 \pm 0,18$	2,37e-02	$5,85\text{e}-04$
	QP	$1,16\text{e}+03 \pm 2,03\text{e}+02$	✗	$155,00 \pm 0,00$	$2,68\text{e}-01$	$5,89\text{e}-04$
	ISDA	$1,17\text{e}+03 \pm 2,08\text{e}+02$	✗	$154,93 \pm 0,25$	$3,30\text{e}-02$	$5,92\text{e}-04$
	ESVR	$1,66\text{e}+03 \pm 4,70\text{e}+02$	-	$27,03 \pm 22,00$	$1,01\text{e}+01$	$2,09\text{e}-05$
BOS	SMO	$2,55\text{e}+01 \pm 8,10\text{e}+00$	✗	$404,90 \pm 0,31$	1,29e-01	$5,80\text{e}-04$
	QP	$2,54\text{e}+01 \pm 8,09\text{e}+00$	✗	$405,00 \pm 0,00$	$3,72\text{e}+00$	$6,42\text{e}-04$
	ISDA	$2,64\text{e}+01 \pm 8,52\text{e}+00$	✗	$316,90 \pm 19,04$	$3,85\text{e}-01$	$8,75\text{e}-04$
	ESVR	$3,82\text{e}+01 \pm 1,29\text{e}+01$	-	$332,77 \pm 54,96$	$2,16\text{e}+01$	$6,63\text{e}-05$
MOT	SMO	$2,28\text{e}+03 \pm 5,52\text{e}+02$	✓	$105,97 \pm 0,18$	1,99e-02	$5,58\text{e}-04$
	QP	$2,29\text{e}+03 \pm 5,64\text{e}+02$	✓	$106,00 \pm 0,00$	$1,60\text{e}-01$	$5,09\text{e}-04$
	ISDA	$2,31\text{e}+03 \pm 5,68\text{e}+02$	✓	$104,37 \pm 1,27$	$2,70\text{e}-02$	$5,68\text{e}-04$
	ESVR	$2,17\text{e}+03 \pm 4,44\text{e}+02$	-	$65,90 \pm 7,24$	$2,65\text{e}+00$	$6,78\text{e}-06$
AIL	SMO	$4,68\text{e}-08 \pm 8,24\text{e}-09$	✗	$13,87 \pm 1,87$	$1,24\text{e}-01$	$6,94\text{e}-04$
	QP	-	-	-	-	-
	ISDA	$4,37\text{e}-08 \pm 4,26\text{e}-09$	✓	$13,70 \pm 1,88$	1,13e-01	$6,56\text{e}-04$
	ESVR	$3,37\text{e}-08 \pm 4,10\text{e}-09$	-	$406,67 \pm 80,66$	$5,78\text{e}+00$	$2,48\text{e}-03$
ELE	SMO	$2,51\text{e}-06 \pm 1,32\text{e}-07$	✗	$1441,80 \pm 190,34$	$4,57\text{e}-01$	$6,63\text{e}-03$
	QP	-	-	-	-	-
	ISDA	$2,11\text{e}-06 \pm 7,81\text{e}-08$	✓	$556,53 \pm 84,15$	3,16e-01	$3,19\text{e}-03$
	ESVR	$2,16\text{e}-06 \pm 8,03\text{e}-08$	-	$310,00 \pm 114,77$	$5,24\text{e}+00$	$3,19\text{e}-03$

Fonte: autor.

Ao se observar a Tabela 3, que apresenta os resultados com *kernel* linear, percebe-se que a proposta ESVR obteve desempenho inferior em termos de MSE aos demais métodos em 3/8 dos conjuntos de dados (AUT, BCR e BOS). Na base BCR o valor do MSE é superior, mas ainda próximo dos demais, porém o teste não indicou equivalência estatística devido elevado valor de desvio padrão da proposta. Por outro lado, percebe-se que nossa proposta obteve melhor desempenho em 4/8 dos conjuntos de dados (CPU, SER, MOT e AIL).

Com relação ao conjunto de dados ELE, para 7613 padrões de treinamento, ESVR obteve desempenho superior ao SMO e equivalente ao ISDA. Já no conjunto de dados AIL com 5703 padrões no treinamento, ESVR obteve desempenho superior aos demais. Vale ressaltar ainda a elevada esparsidade alcançada pelo ESVR para o conjunto de dados ELE, sendo tal resultado equivalente ao obtido pelo ISDA que utiliza em média 310 vetores suporte, enquanto o SMO utiliza em média 1450 vetores suporte. Não foi possível utilizar o método de treinamento QP devido a demanda computacional exigida pelo método.

Com relação a esparsidade nos resultados com *kernel* linear, a proposta ESVR resultou em mais esparsidade em 6/8 dos conjuntos de dados (AUT, CPU, SER, BCR, MOT e ELE). Tal resultado pode ser melhor visualizado na Figura 14, onde os resultados das 30 realizações para os conjuntos de dados SER, AIL e ELE são apresentados relativo a cada um dos métodos de treinamento. A relação do gráfico é dado pelas métricas MSE e também pelo número de vetores suporte (#SV), de tal maneira que quanto mais próximo de (0,0) os agrupamentos estiverem, melhor são os valores de MSE e menos vetores suportes foram necessários.

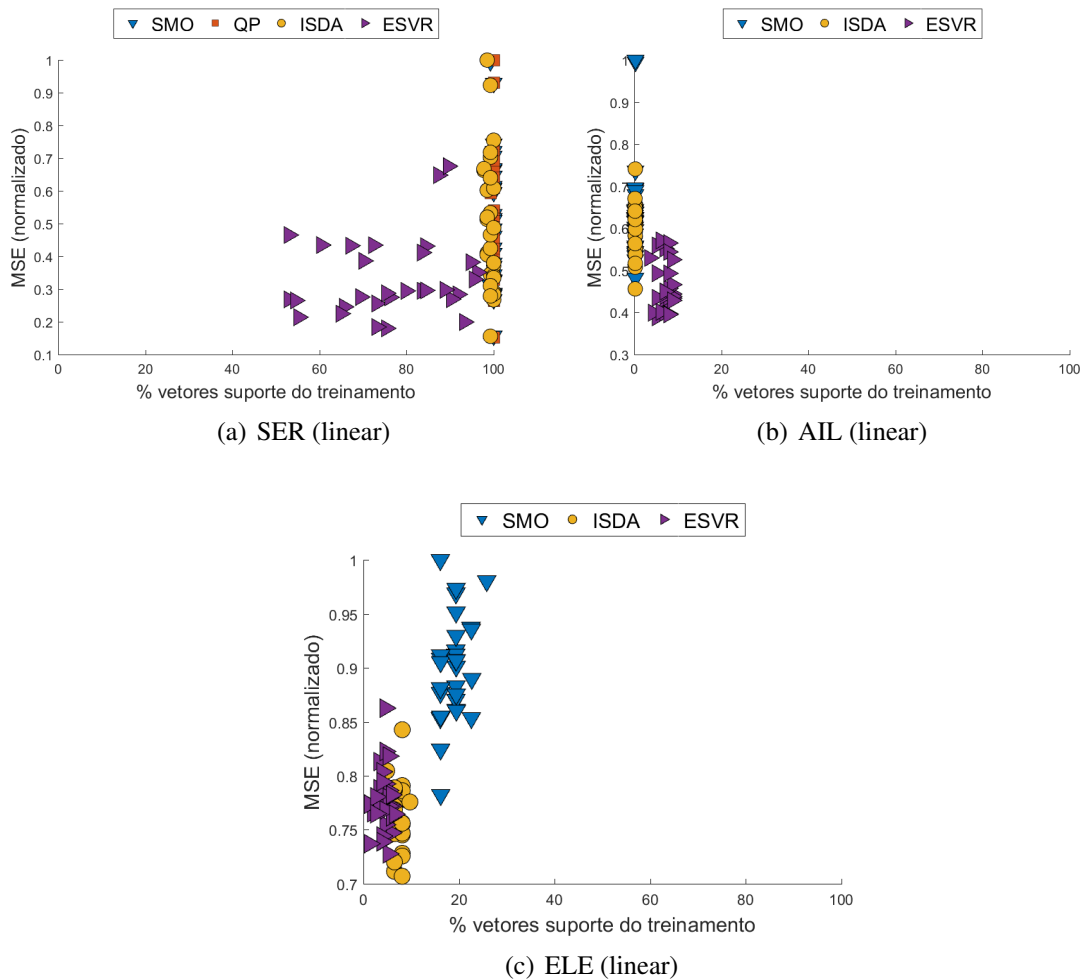


Figura 14: Relação MSE e SV nos conjuntos SER, AIL e ELE com *kernel* linear.

Como mencionado anteriormente, uma vez que o objetivo é a apresentação da informação

graficamente, os valores foram ajustados para melhor visualização. Portanto, o número de vetores suporte é mostrado na Figura 14 como uma porcentagem dada em relação ao conjunto de treinamento e o MSE foi normalizado entre 0 e 1. Ainda na figura é possível ver que a proposta ESVR quando aplicado ao conjunto de dados ELE obteve um MSE equivalente ao ISDA e mais esparsidade que o mesmo. Além disso, ambos foram superiores ao algoritmo SMO. No conjunto de dados AIL, a nossa proposta superou os demais métodos de treinamento com relação ao MSE. Já em relação a esparsidade, a proposta foi a menos esparsa. Destaca-se que o ISDA e SMO precisaram de apenas $\approx 13,9$ vetores em média. Como mencionado, não foi possível executar o método QP nestes conjuntos devido ao custo computacional.

Ao analisar a Tabela 4, que apresenta os resultados com *kernel* gaussiano, pode-se observar que a proposta ESVR obteve resultados melhores em relação aos obtidos anteriormente.

Tabela 4: Resultados da média das 30 realizações com *kernel* gaussiano.

Base	Método	MSE	TF (MSE)	#SV	TR (s)	TE (s)
AUT	SMO	$6,34e+06 \pm 4,26e+06$	✓	$66,70 \pm 6,17$	1,22e-02	$4,44e-04$
	QP	$6,34e+06 \pm 4,26e+06$	✓	$66,77 \pm 6,21$	$1,70e-01$	$4,19e-04$
	ISDA	$7,11e+06 \pm 4,21e+06$	✗	63,90 $\pm$ 4,12	$2,08e-02$	$4,49e-04$
	ESVR	5,39e+06 $\pm$ 2,14e+06	-	$100,43 \pm 25,48$	$4,08e+00$	1,39e-05
CPU	SMO	6,15e+03 $\pm$ 6,75e+03	✓	$51,20 \pm 7,90$	1,03e-02	$4,22e-04$
	QP	$6,15e+03 \pm 6,76e+03$	✓	$51,07 \pm 7,66$	$2,67e-01$	$4,08e-04$
	ISDA	$6,49e+03 \pm 7,69e+03$	✓	44,67 $\pm$ 5,98	$1,28e-02$	$4,24e-04$
	ESVR	$6,77e+03 \pm 8,42e+03$	-	$131,27 \pm 21,73$	$4,07e+00$	1,73e-05
SER	SMO	$4,54e-01 \pm 3,75e-01$	✓	$58,30 \pm 16,64$	1,08e-02	$4,12e-04$
	QP	4,54e-01 $\pm$ 3,75e-01	✓	$60,17 \pm 17,80$	$1,34e-01$	$4,03e-04$
	ISDA	$4,60e-01 \pm 3,91e-01$	✓	56,27 $\pm$ 17,07	$1,36e-02$	$4,30e-04$
	ESVR	$5,06e-01 \pm 3,87e-01$	-	$89,97 \pm 15,91$	$5,71e+00$	9,51e-06
BCR	SMO	$1,20e+03 \pm 1,60e+02$	✓	$155,00 \pm 0,00$	1,21e-02	$5,43e-04$
	QP	$1,20e+03 \pm 1,60e+02$	✗	$155,00 \pm 0,00$	$1,18e-01$	$4,86e-04$
	ISDA	$1,21e+03 \pm 1,81e+02$	✗	$154,90 \pm 0,31$	$1,25e-02$	$4,99e-04$
	ESVR	1,07e+03 $\pm$ 1,83e+02	-	100,97 $\pm$ 22,17	$7,28e+00$	1,17e-05
BOS	SMO	$1,02e+01 \pm 4,15e+00$	✗	$278,87 \pm 13,46$	4,60e-02	$8,25e-04$
	QP	$1,02e+01 \pm 4,16e+00$	✗	$279,20 \pm 13,54$	$2,12e+00$	$5,87e-04$
	ISDA	9,86e+00 $\pm$ 3,63e+00	✗	274,93 $\pm$ 11,31	$1,16e-01$	$5,86e-04$
	ESVR	$1,57e+01 \pm 5,98e+00$	-	$327,53 \pm 60,06$	$4,69e+01$	6,34e-05
MOT	SMO	$5,81e+02 \pm 1,88e+02$	✓	$106,00 \pm 0,00$	1,34e-02	$5,31e-04$
	QP	$5,81e+02 \pm 1,88e+02$	✓	$106,00 \pm 0,00$	$1,38e-01$	$5,24e-04$
	ISDA	$5,81e+02 \pm 1,88e+02$	✓	$105,93 \pm 0,25$	$1,68e-02$	$5,51e-04$
	ESVR	5,74e+02 $\pm$ 1,71e+02	-	62,03 $\pm$ 22,63	$1,79e+00$	9,65e-06
AIL	SMO	$6,63e-08 \pm 4,57e-09$	✓	$212,00 \pm 0,00$	$1,39e-01$	$3,62e-03$
	QP	-	-	-	-	-
	ISDA	3,80e-08 $\pm$ 2,99e-09	✓	106,00 $\pm$ 0,00	1,22e-01	1,69e-03
	ESVR	$4,86e-08 \pm 6,46e-09$	-	$435,00 \pm 70,89$	$4,40e+00$	$2,61e-03$
ELE	SMO	$2,29e-06 \pm 1,10e-07$	✗	$3737,67 \pm 147,92$	$1,19e+00$	$1,01e-01$
	QP	-	-	-	-	-
	ISDA	2,09e-06 $\pm$ 9,78e-08	✓	$2925,40 \pm 183,05$	1,13e+00	$5,19e-02$
	ESVR	$2,48e-06 \pm 1,38e-07$	-	453,33 $\pm$ 64,24	$4,76e+00$	4,89e-03

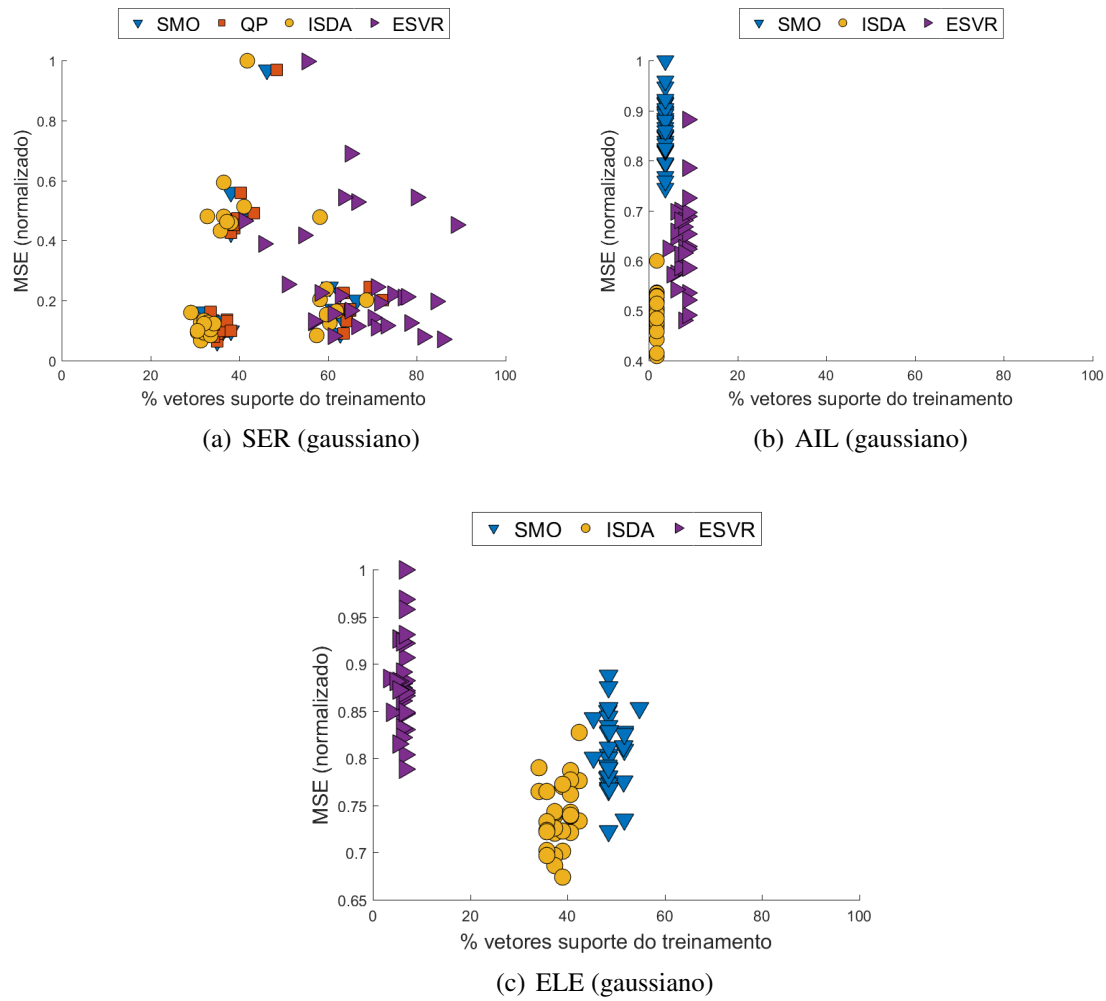
Fonte: autor.

Não só melhor desempenho em relação a quando aplicado com *kernel* linear como, principalmente, em relação aos demais métodos de treinamento. Como pode ser visto, o teste estatístico de Friedman indicou relação de equivalência com todos os demais métodos em 7/8 dos conjuntos de dados utilizados, sendo que apenas para o conjunto de dados BOS, a proposta obteve desempenho inferior em ambos os resultados com *kernel* linear e gaussiano. Dentre esses conjuntos, tem-se o BCR, no qual o desempenho havia sido inferior com o *kernel* linear, neste resultado tem-se que ESVR obteve desempenho superior aos demais, visto que apenas o algoritmo SMO foi dado como equivalente a proposta para este conjunto.

Com relação aos conjuntos de dados AIL e ELE, cujo tamanho é superior a 7000, percebe-se que mesmo que ESVR trabalhe apenas com 500 padrões do conjunto como mencionado na metodologia, pode-se perceber que o seu desempenho permanece adequado, visto que ESVR com relação ao conjunto de dados ELE, alcançou mais esparsidade e mesmo não obtendo o melhor resultado com relação ao MSE, seu desempenho foi equivalente ao ISDA. Obteve ainda, desempenho superior, com *kernel* linear e em termos de MSE, a todos os demais métodos no conjunto de dados AIL, e quando foi utilizado o *kernel* gaussiano seu desempenho foi equivalente ao ISDA. Vale ressaltar que, como mencionado no Capítulo 3, o método de treinamento ISDA foi desenvolvido especialmente para grandes conjuntos de dados. Este comportamento pode ser comprovado, visto que nos demais conjuntos de dados, o método com base no ISDA se mantém com desempenho inferior a SMO e QP, já nos conjuntos maiores AIL e ELE, o método ISDA obtém 1º e 2º melhor desempenho, no tocante ao MSE.

A Figura 15 apresenta a relação do MSE com o número de vetores suporte nos conjuntos SER, AIL e ELE, para *kernel* gaussiano. Pode-se notar que os valores de MSE para ambos os métodos estão praticamente em um mesmo intervalo. Adicionalmente, na Figura 16 são apresentados os gráficos da relação do MSE com vetores suporte das realizações sobre os conjuntos de dados AUT, CPU, SER, BCR, BOS e MOT. Nesta figura são apresentados os resultados para as 30 realizações com *kernel* linear e gaussiano em cada um dos conjuntos de dados, sendo possível perceber a similaridade em termos de MSE e número de vetores suporte a partir da percepção dos agrupamentos formados pelos resultados dos métodos comparados.

ESVR obteve realizações com número de vetores suporte variados ao passo que, os demais métodos mantiveram seu número de vetores suporte com pouquíssima variação dentre as 30 realizações. Esta característica da proposta, deve-se aos algoritmos genéticos, bem como ao algoritmo de ajuste, que em seu processo busca reduzir a quantidade de vetores suporte ao deduzir valores dos multiplicadores de Lagrange.

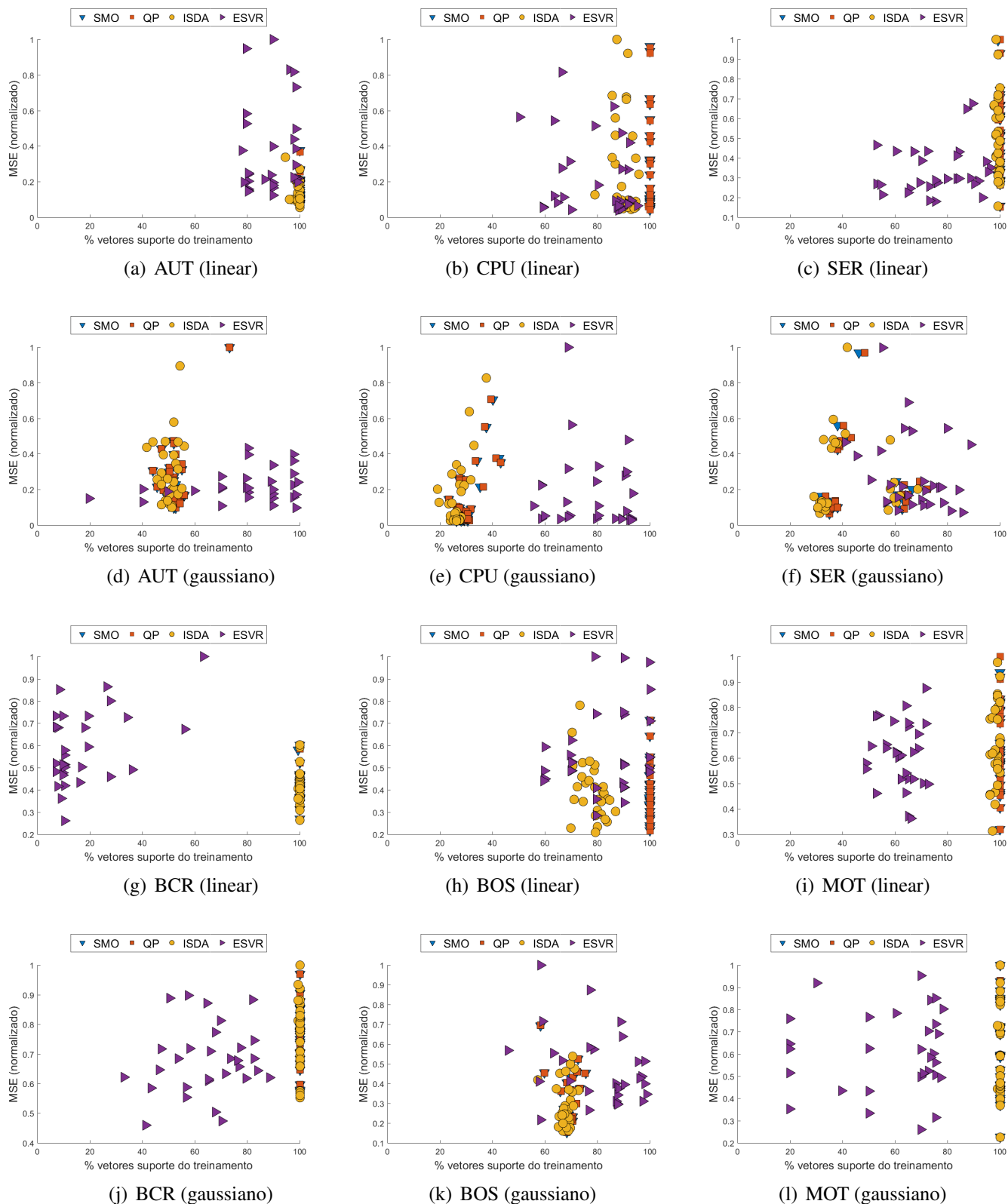
Figura 15: Relação MSE e SV para os conjuntos de dados SER, AIL e ELE com *kernel* gaussiano.

Fonte: autor.

Ao se observar o resultado para ambos os *kernels* é possível notar, pela análise dos agrupamentos, que o desempenho dos métodos SMO, QP e ISDA são semelhantes. Este comportamento é mais evidente no conjunto de dados SER com *kernel* gaussiano, onde se formaram 3 agrupamentos dentre as realizações. A semelhança supracitada não ocorre quando se trata dos conjuntos de dados grandes (AIL e ELE), neste caso o algoritmo baseado em ISDA obtém melhor desempenho devido as suas características já discutidas anteriormente.

A proposta ESVR se manteve em uma faixa semelhante para o número de vetores suporte independentemente do *kernel* utilizado. Já nos demais métodos houve uma redução do número de vetores suporte, quando o *kernel* foi substituído pelo gaussiano. Ou seja, ao utilizar um *kernel* linear foi necessário uma maior quantidade de vetores suporte, o que não acontece com a nossa proposta. O conjunto de dados BCR foi o único que não seguiu este comportamento.

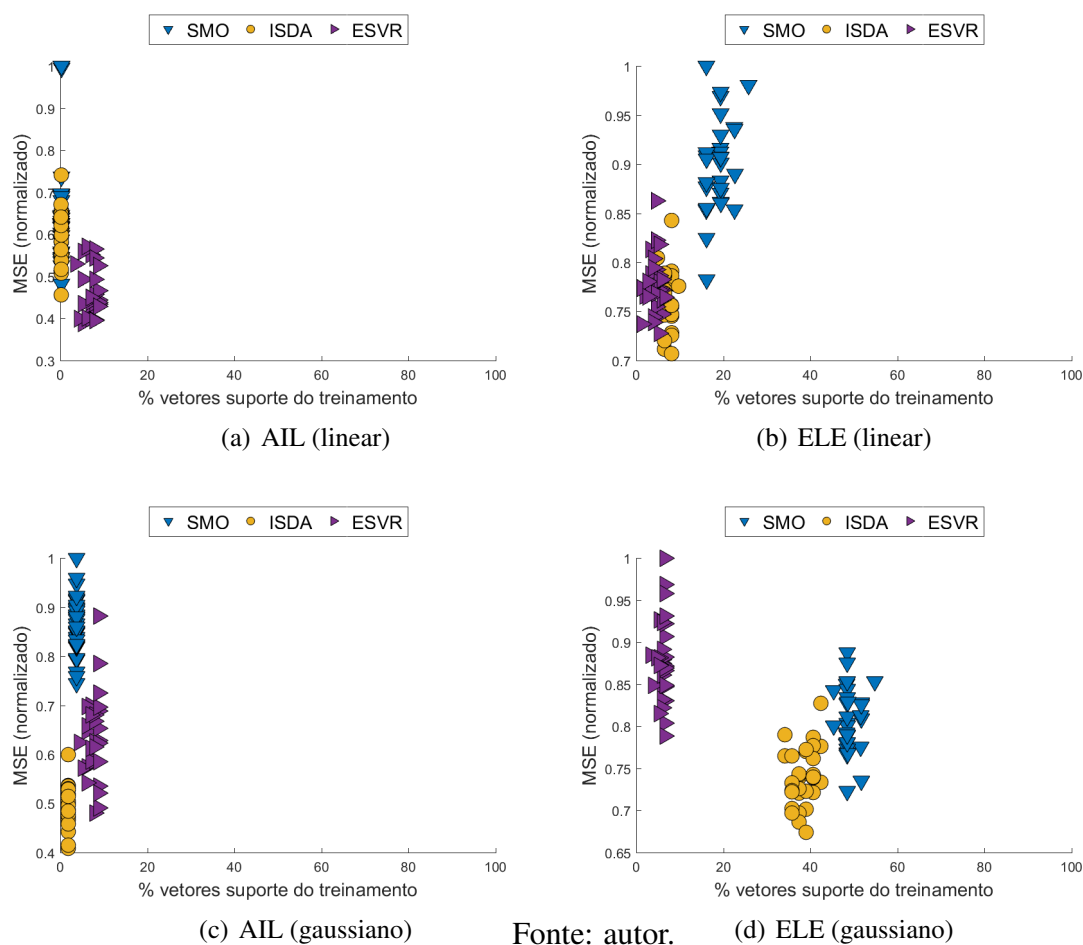
Figura 16: Relação MSE e SV para os conjuntos de dados AUT, CPU, SER, BCR, BOS e MOT com *kernel* linear e gaussiano.



Fonte: autor.

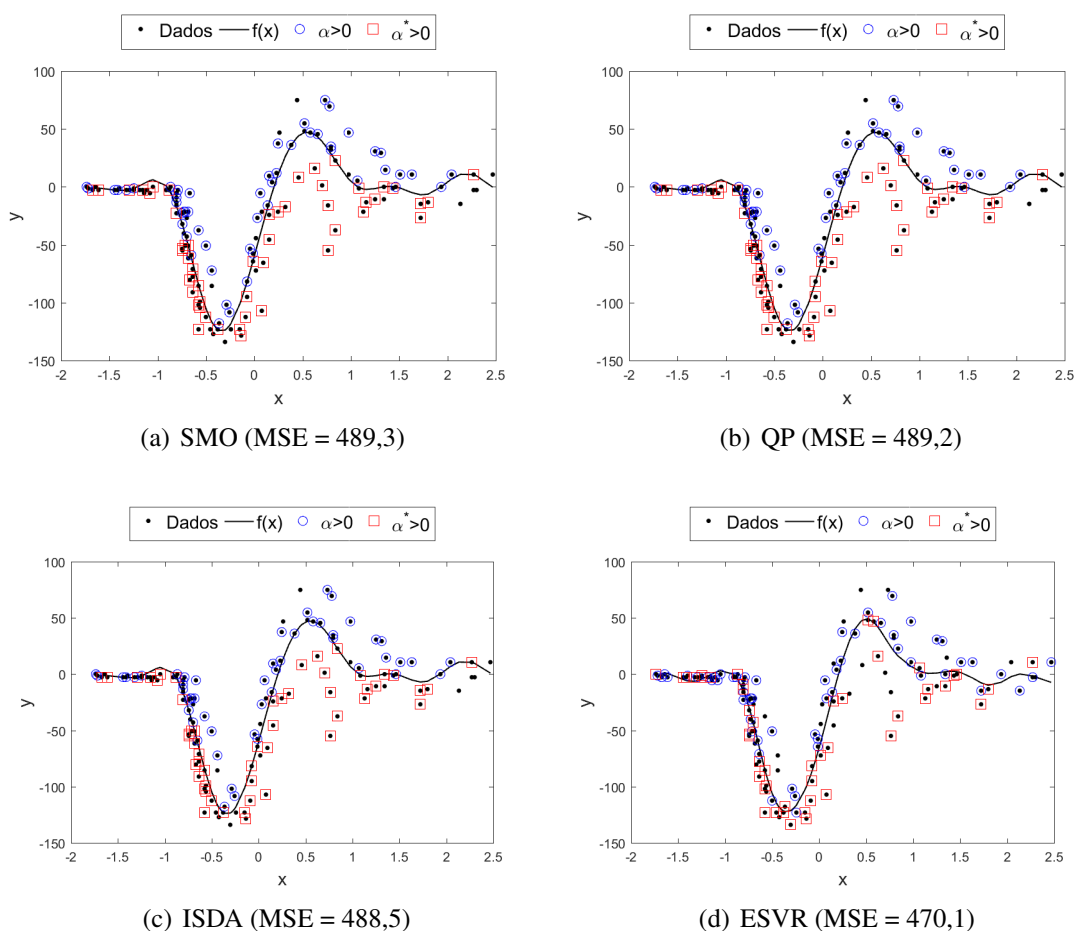
A Figura 17 apresenta as mesmas informações da Figura 16, agora sobre os conjuntos de dados AIL e ELE. Pode-se observar que, no conjunto de dados AIL, todos os métodos alcançaram elevada esparsidade. O desempenho, em geral para ambos os *kernels* foram semelhantes, diferindo no conjunto de dados ELE, no qual se percebe um número mais elevado de vetores suporte para os métodos SMO e ISDA, enquanto a nossa proposta ESVR alcançou maior esparsidade para ambos os *kernels*. Em geral, podemos concluir que a nossa proposta obteve bons resultados para conjuntos de dados maiores.

Figura 17: Relação MSE e SV nos conjuntos de dados AIL e ELE com *kernel* linear e gaussiano.



Na Figura 18 são apresentados os resultados com a utilização do *kernel* gaussiano para o conjunto de dados MOT, sobre o qual a proposta ESVR obteve melhor desempenho tanto em termos de MSE quanto vetores suporte. Nesta figura é mais evidente que o comportamento dos métodos foram similares entre si tanto em relação ao valor de MSE quanto no que se refere a quais padrões do treinamento são dados como vetores suporte. Além disso, na figura é demarcado quais vetores suporte possuem os multiplicadores de Lagrange $\alpha > 0$ e quais $\alpha^* > 0$, os quais também são similares dentre os métodos comparados.

Figura 18: Conjunto de dados MOT com os métodos SMO, QP, ISDA e ESVR com *kernel* gaussiano. Círculos vazados são os padrões cujo $\alpha_i > 0$ e quadrados vazados se $\alpha_i^* > 0$.



Fonte: autor.

No gráfico em barras da Figura 19 (a) e (b), é apresentado o valor de MSE médio para as 30 realizações. Os valores foram normalizados através da técnica min-max para que fosse possível apresentar os valores de desempenho com relação ao MSE sobre todos os conjuntos de dados em um único gráfico e desta forma obter uma visão geral dos resultados. Ao observar o gráfico, pode-se confirmar novamente a semelhança de desempenho dentre os métodos comparados. Na Figura 19 (a) é possível observar os métodos QP, SMO e ISDA obtiveram melhor desempenho que a nossa proposta para os conjuntos de dados AUT, BCR e BOS, por outro lado, a nossa proposta ESVR obteve melhor desempenho ou equivalente para os demais conjuntos de dados CPU, SER, MOT, AIL e ELE. Com

relação ao número de vetores suporte, o gráfico em barras da Figura 19 (c) e (d) apresenta a média do número de vetores suporte das 30 realizações. Percebe-se que a nossa proposta ESVR é evidentemente mais esparsa que os métodos comparados para ambos os *kernel* linear e gaussiano.

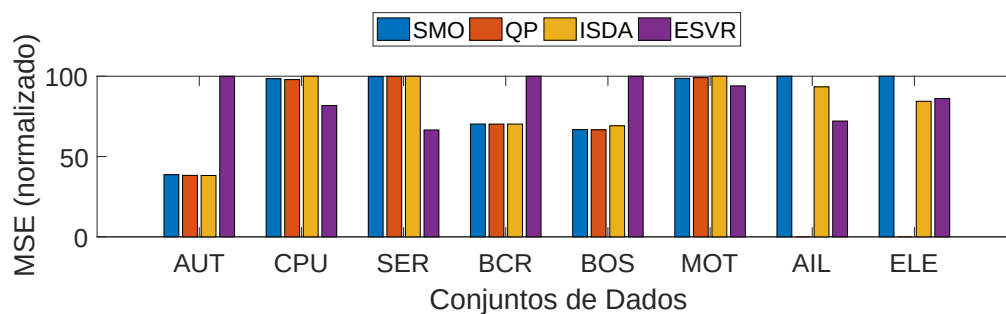
Com relação ao custo computacional apresentado nas tabelas 3 e 4, percebe-se que a nossa proposta possui o maior custo computacional em relação aos demais métodos de treinamento, mas este custo permanece na mesma unidade (segundos). Dentre os resultados em termos de custo computacional para o *kernel* linear, o maior custo em média obtido pela nossa proposta ESVR foi de 21 segundos, ao passo que os demais métodos realizaram o treinamento em menos de 3 segundos. Para o *kernel* gaussiano, o custo de treinamento da nossa proposta ESVR foi em média 7,28 segundos desconsiderando o conjunto de dados BOS, no qual o custo médio foi de 46 segundos. O custo relacionado ao conjunto de dados BOS foi mencionado separadamente, visto que o seu custo de treinamento (46 segundos) foi elevado quando comparado a média do custos sobre os demais 7 conjuntos de dados (7,28 segundos), logo inseri-lo na média geral de custo de treinamento da nossa proposta não refletiria o real custo de treinamento. O custo computacional na etapa de treinamento, incluindo o conjunto de dados BOS, para os demais métodos, foi em média menos de 4 segundos. Em contrapartida, com relação ao tempo de teste, observa-se que a nossa proposta ESVR possui o menor custo computacional em relação aos demais métodos de treinamento. Os valores estão na mesma unidade e são menores que 0,0001 segundo.

É sabido que em algoritmos genéticos, o custo da evolução de diversas soluções em paralelo, assim como o tamanho dessas soluções, pode demandar mais custo computacional. A modelagem do problema de otimização quadrática inerente à SVRs para o arcabouço de algoritmos genéticos envolve a otimização de dois multiplicadores de Lagrange por padrão de treinamento, diferentemente de SVC que necessita otimizar apenas um multiplicador de Lagrange por padrão do treinamento. Logo, inicialmente, a representação genética da nossa solução seria dada por um indivíduo de tamanho $2l$, ou seja, o dobro da tamanho do conjunto de treinamento. Após algumas análises apresentadas no Capítulo 4, foi possível manter o indivíduo do tamanho do conjunto de treinamento. Porém, mesmo assim, foi necessário reduzir o conjunto de treinamento para os conjuntos de dados grandes para viabilizar a aplicação de algoritmos genéticos, tal procedimento não prejudicou o desempenho da proposta como já apresentado.

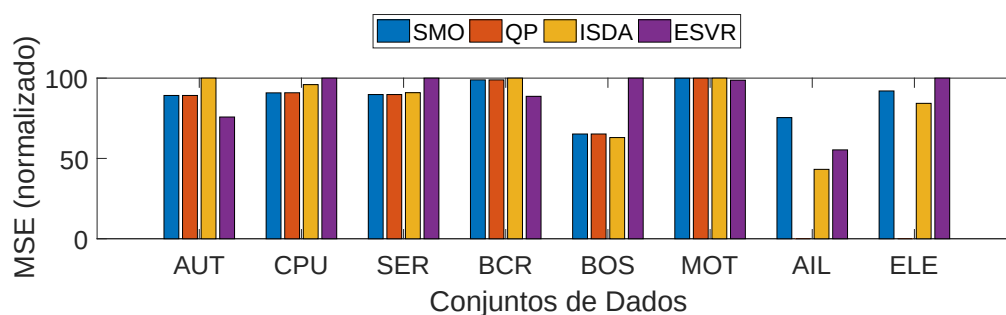
Antes da realização dos experimentos finais, foi realizado um levantamento do custo computacional da proposta a fim de identificar quais processos internos poderiam ser otimizados para que houvesse uma redução do seu custo computacional. Foi verificado então que, armazenar a matriz de *kernel* acarreta na redução do tempo de treinamento. Vale ressaltar que os algoritmos SMO e os baseados em ISDA utilizam a abordagem de decomposição do problema de otimização quadrática assim como estratégias de seleção do conjunto de trabalho. Nossa proposta, diferentemente, otimiza o problema de otimização como um todo, e mesmo assim, como verificado a partir da análise dos resultados, a nossa proposta obteve resultados competitivos.

No Capítulo 3 foi apresentado o *ϵ -evolutionary support vector regression*, a única abordagem

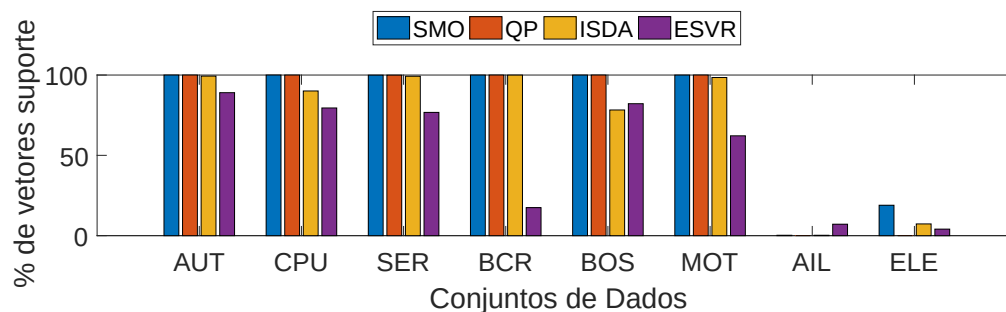
Figura 19: Relação do MSE normalizado e relação de vetores suporte conjunto/método com *kernel* linear e gaussiano.



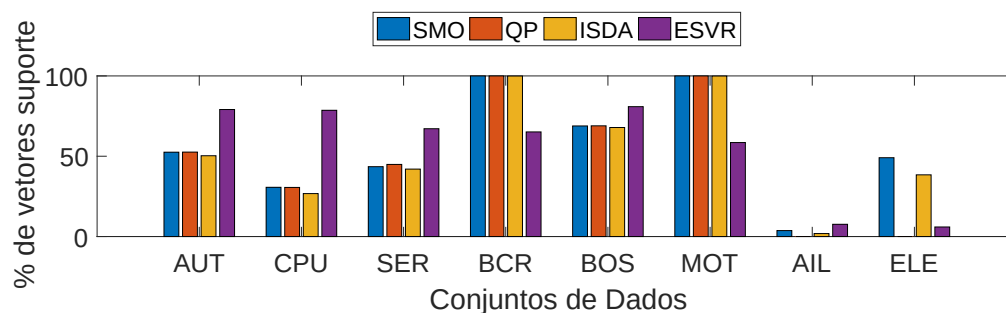
(a) MSE normalizado (*kernel* linear).



(b) MSE normalizado (*kernel* gaussiano).



(c) Número de SVs (*kernel* linear).



(d) Número de SVs (*kernel* gaussiano).

Fonte: autor.

encontrada até o momento na literatura que aplica algoritmos genéticos no treinamento de SVRs. No entanto, a abordagem apresentada soluciona o problema de otimização quadrática em sua formulação

primal que difere da formulação dual solucionada nesta dissertação. A abordagem supracitada só pode ser aplicada com *kernel* linear por se tratar da formulação primal, limitando a capacidade do regressão da abordagem. Além disso, realizou seus experimentos somente sobre 10 realizações para apenas um conjunto de dados BOS e não foram realizadas comparações com os métodos QP, ISDA, SMO. O conjunto de dados BOS, foi o único conjunto de dados apresentado aqui, para o qual a nossa proposta não obteve resultado satisfatório, o que não invalida a nossa proposta tendo em vista os bons resultados obtidos nos demais conjuntos de dados.

5.2.1 Conclusão

Neste capítulo é apresentada a metodologia utilizada para validação da proposta, bem como as características dos conjuntos de dados utilizados. Os resultados obtidos pela proposta ESVR foram comparados com os resultados dos outros quatro algoritmos de aprendizagem descritos no Capítulo 3. As métricas de avaliação dos métodos consistem no valor do MSE, número de vetores suporte, tempo de treinamento e teste, e o teste estatístico. Em resumo, os resultados indicam que a nossa proposta ESVR se apresenta como uma abordagem promissora, visto que obteve desempenho superior ou equivalente em praticamente todos os conjuntos de dados utilizados, sendo inferior aos demais métodos somente para o conjunto de dados BOS. Além do mais, a proposta também se mostra competitiva em relação ao número de vetores suporte.

CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho foi proposta uma nova abordagem para a resolução do problema de otimização quadrática na sua formulação dual inerente à Máquinas de Vetores Suporte para Regressão (SVR) utilizando Algoritmos Genéticos. A proposta inclui a modelagem de SVR para o arcabouço de Algoritmos Genéticos, o que inclui a representação genética, os operadores de mutação e reprodução adequados para o contexto do problema, assim como a função de aptidão com base, na já supracitada, formulação dual do problema. A partir disto, ao processar o indivíduo ótimo oriundo da população final (evoluída), obtivemos como resultado um satisfatório conjunto de multiplicadores de Lagrange que é a solução do problema dual.

Para avaliar a nossa proposta, foram realizadas simulações computacionais em oito bases de dados reais disponíveis no repositório de aprendizado de máquinas da Universidade da Califórnia em Irvine, a saber *Auto price*, *Servo*, *Breast cancer wisconsin*, *CPU*, *Boston*, *Motorcycle*, *Ailerons* e *Elevators*. Os resultados experimentais indicam que a abordagem proposta provê resultados competitivos quando comparado aos outros métodos de otimização para o problema dual em Máquinas de Vetor Suporte para Regressão, a saber *Sequential Minimal Optimization*, Programação Quadrática e o *Iterative single data algorithm*. Isto foi confirmado pelas equivalências obtidas através de testes estatísticos.

Ademais, uma presente limitação da proposta é que a mesma necessita reduzir o conjunto de treinamento quando o mesmo ultrapassa mil padrões, nos casos dos conjuntos de dados *Ailerons* e *Elevators*, outros métodos também passam por esse tipo de limitação. Entretanto, mesmo havendo realizado esta redução utilizando um método de agrupamento, denominado *k-medoids*, constatou-se que a proposta manteve um bom desempenho. A proposta alcançou valores de erro quadrático médio inferiores aos demais métodos em alguns casos, e quando não o alcançou, os demais métodos foram dados como equivalentes a proposta pelo teste estatístico.

Acerca dos custos computacionais, é sabido que a nossa abordagem demanda mais processamento e tempo de execução durante o treinamento, principalmente devido ao cômputo da matriz de kernel, e o tamanho dessa matriz está diretamente relacionado ao tamanho do conjunto de treinamento. No entanto, foi apresentado que a proposta obteve menor custo computacional quando observado o tempo de teste. De fato, os menores tempos de teste foram obtidos pela nossa abordagem, o que caracteriza uma vantagem, dado que o processo de treinamento só acontece uma única vez.

6.1 Contribuições deste Trabalho

As contribuições apresentadas nesta dissertação estão relacionadas aos temas: Máquinas de Vetores Suporte para Regressão e Algoritmos Genéticos. O Capítulo 3 fornece ao leitor uma compreensão sobre Máquinas de Vetores Suporte para Regressão e seus métodos de treinamento, o Capítulo 2 apresenta de forma detalhada os aspectos relacionados a Algoritmos Genéticos. No Capítulo 4 é apresentada a proposta que visa a solução do problema de otimização quadrática na sua formulação dual em Máquina de Vetor Suporte para regressão. Além disto, neste mesmo capítulo, são apresentados a modelagem para a representação genética, os operadores de mutação e reprodução, algoritmo de ajuste e a função de aptidão com base nos multiplicadores de Lagrange. No Capítulo 5 é apresentada a metodologia utilizada para avaliar a proposta, desde quais conjuntos de dados foram utilizadas nos experimentos ao parâmetros de configuração dos mesmos, da proposta e dos métodos de treinamento mais utilizados na literatura. Por fim, no Capítulo 5, os resultados são apresentados por meio de tabelas e gráficos, os mesmos são analisados e discutidos a fim de avaliar a proposta deste trabalho.

Em relação ao desempenho, a proposta foi avaliada em termos do erro quadrático médio, o qual é amplamente utilizado para avaliar problemas de regressão. Além do erro quadrático médio, foi avaliado também a esparsidade, uma característica bastante desejada quando se trabalha com máquinas de vetores suporte. A proposta então foi avaliada no Capítulo 5, e os resultados obtidos evidenciam que a proposta dessa dissertação está de acordo com os objetivos inicialmente definidos.

6.2 Trabalhos Futuros

Pode-se sugerir aplicar os três critérios de convergências apresentados no Capítulo 3 e avaliar novos resultados com relação ao uso destes critérios. Essa sugestão é devido ao fato de que a função de aptidão é dada em função da resolução do próprio problema de otimização quadrática, a qual é dada em termos de maximização. Outra sugestão é explorar a população final de soluções a fim de não só encontrar os multiplicadores de Lagrange, mas também, quem sabe, escolher quais os vetores suporte resultam em melhor desempenho.

Outra sugestão consiste na inicialização da população com base em alguma heurística. É conhecido que os custos computacionais em Algoritmos Genéticos podem ser reduzidos quando se adota uma população bem diversificada. Adicionalmente, os parâmetros de configuração de Algoritmos Genéticos estão diretamente ligados ao desempenho e capacidade de convergência do algoritmo, logo uma melhor análise destes parâmetros se faz necessária.

Por fim, pretende-se explorar abordagens diferentes nos aspectos inerentes a algoritmos genéticos, como por exemplo, em termos de configuração de seus parâmetros e dos operadores genéticos, a fim de elevar a qualidade dos indivíduos no que compete a obter resultados melhores.

Referências

- ARTHUR, David; VASSILVITSKII, Sergei. k-means plus plus : The Advantages of Careful Seeding. In: ACM SIGACT; SIAM. **PROCEEDINGS OF THE EIGHTEENTH ANNUAL ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS**. 3600 UNIV CITY SCIENCE CENTER, PHILADELPHIA, PA 19104-2688 USA: SIAM, 2007. p. 1027–1035. ISBN 978-0-898716-24-5. 18th ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, JAN 07-09, 2007.
- BACHE, K.; LICHMAN, M. **UCI Machine Learning Repository**. 2013. Disponível em: <<http://archive.ics.uci.edu/ml>>.
- BÄCK, Thomas; FOGEL, David B; MICHALEWICZ, Zbigniew. Handbook of evolutionary computation. **New York: Oxford**, 1997.
- BANZHAF, Wolfgang; NORDIN, Peter; KELLER, Robert E; FRANCONI, Frank D. **Genetic programming: an introduction**. [S.l.]: Morgan Kaufmann San Francisco, 1998.
- BEN-HUR, Asa; WESTON, Jason. A user's guide to support vector machines. **Data mining techniques for the life sciences**, Springer, p. 223–239, 2010.
- BISHOP, C. Pattern recognition and machine learning (information science and statistics), 1st edn. 2006. corr. 2nd printing edn. **Springer, New York**, 2007.
- BOSER, Bernhard E; GUYON, Isabelle M; VAPNIK, Vladimir N. A training algorithm for optimal margin classifiers. In: ACM. **Proceedings of the fifth annual workshop on Computational learning theory**. [S.l.], 1992. p. 144–152.
- CARDOSO, Fabbryccio ACM. **Algoritmos Genéticos para Interpolação Não-Linear de Imagem e Decodificação de Códigos Lineares**. Mestrado em Engenharia Elétrica — Universidade Estadual de Campinas, 1998.
- CERVANTE, Liam; XUE, Bing; ZHANG, Mengjie; SHANG, Lin. Binary particle swarm optimisation for feature selection: A filter based approach. In: IEEE. **Evolutionary Computation (CEC), 2012 IEEE Congress on**. [S.l.], 2012. p. 1–8.
- CHEN, Kuan-Yu; WANG, Cheng-Hua. Support vector regression with genetic algorithms in forecasting tourism demand. **Tourism Management**, Elsevier, v. 28, n. 1, p. 215–226, 2007.
- CHEN, Pai-Hsuen; FAN, Rong-En; LIN, Chih-Jen. A study on smo-type decomposition methods for support vector machines. **IEEE Trans. Neural Networks**, v. 17, n. 4, p. 893–908, 2006.
- CHERKASSKY, V; MA, YQ. Practical selection of SVM parameters and noise estimation for SVM regression. **NEURAL NETWORKS**, PERGAMON-ELSEVIER SCIENCE LTD, THE BOULEVARD, LANGFORD LANE, KIDLINGTON, OXFORD OX5 1GB, ENGLAND, 17, n. 1, p. 113–126, JAN 2004. ISSN 0893-6080.
- CORTES, Corinna; VAPNIK, Vladimir. Support-vector networks. **Machine learning**, Springer, v. 20, n. 3, p. 273–297, 1995.

- COVER, Thomas M. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. **IEEE transactions on electronic computers**, IEEE, n. 3, p. 326–334, 1965.
- DAVIS, Lawrence. Applying adaptive algorithms to epistatic domains. In: **IJCAI**. [S.l.: s.n.], 1985. v. 85, p. 162–164.
- DAVIS, Lawrence. **"Handbook of genetic algorithms"**. [S.l.]: "Van Nostrand Reinhold", 1991.
- DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **JOURNAL OF MACHINE LEARNING RESEARCH**, MICROTOME PUBLISHING, 31 GIBBS STREET, BROOKLINE, MA 02446 USA, 7, p. 1–30, JAN 2006. ISSN 1532-4435.
- DIAS, Madson Luiz Dantas; ROCHA NETO, Ajalmar R. Evolutionary support vector machines: A dual approach. In: **IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, July 24-29, 2016**. [s.n.], 2016. p. 2185–2192. Disponível em: <<http://dx.doi.org/10.1109/CEC.2016.7744058>>.
- DUMITRESCU, D; PREUSS, Mike; STOEAN, Catalin; STOEAN, Ruxandra. Evolutionary support vector machines and their application for classification. 2006.
- EUBANK, Randall L. **Nonparametric regression and spline smoothing**. [S.l.]: CRC press, 1999.
- FAN, Rong-En; CHEN, Pai-Hsuen; LIN, Chih-Jen. Working set selection using second order information for training support vector machines. **Journal of machine learning research**, v. 6, n. Dec, p. 1889–1918, 2005.
- FOGEL, Lawrence J; OWENS, Alvin J; WALSH, Michael J. Artificial intelligence through simulated evolution. John Wiley, 1966.
- GASCÓN-MORENO, J.; SALCEDO-SANZ, S.; ORTIZ-GARCÍA, E. G.; CARRO-CALVO, L.; SAAVEDRA-MORENO, B.; PORTILLA-FIGUERAS, J. A. A binary-encoded tabu-list genetic algorithm for fast support vector regression hyper-parameters tuning. In: **ISDA, 2011 11th International Conference on**. [S.l.: s.n.], 2011. p. 1253–1257. ISSN 2164-7143.
- GILL, Philip E; MURRAY, Walter; WRIGHT, Margaret H. Practical optimization. Academic press, 1981.
- GOLDBERG, David E; HOLLAND, John H. Genetic algorithms and machine learning. **Machine learning**, Springer, v. 3, n. 2, p. 95–99, 1988.
- GUNN, SR; BROWN, M; BOSSLEY, KM. Network performance assessment for neurofuzzy data modelling. In: Liu, XH and Cohen, P and Berthold, M (Ed.). **ADVANCES IN INTELLIGENT DATA ANALYSIS: REASONING ABOUT DATA**. HEIDELBERGER PLATZ 3, W-1000 BERLIN 33, GERMANY: SPRINGER-VERLAG BERLIN, 1997. (LECTURE NOTES IN COMPUTER SCIENCE, 1280), p. 313–323. ISBN 3-540-63346-4. ISSN 0302-9743. 2nd International Symposium on Advances in Intelligent Data Analysis - Reasoning About Data (IDA-97), LONDON, ENGLAND, AUG 04-06, 1997.
- HAN, Jiawei; PEI, Jian; KAMBER, Micheline. **Data mining: concepts and techniques**. [S.l.]: Elsevier, 2011.
- HOLLAND, John H. **"Adaptation in Natural and Artificial Systems"**. [S.l.: s.n.], 1975.
- HOLLAND, John H. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence**. [S.l.]: MIT press, 1992.

- HOPFIELD, John J. Neural networks and physical systems with emergent collective computational abilities. **Proceedings of the national academy of sciences**, National Acad Sciences, v. 79, n. 8, p. 2554–2558, 1982.
- HUANG, Te-Ming; KECMAN, Vojislav; KOPRIVA, Ivica. **Kernel Based Algorithms for Mining Huge Data Sets: Supervised, Semi-Supervised, and Unsupervised Learning**. [S.l.]: Springer, 2006.
- IJJINA, Earnest Paul; CHALAVADI, Krishna Mohan. Human action recognition using genetic algorithms and convolutional neural networks. **PATTERN RECOGNITION**, ELSEVIER SCI LTD, THE BOULEVARD, LANGFORD LANE, KIDLINGTON, OXFORD OX5 1GB, OXON, ENGLAND, 59, n. SI, p. 199–212, NOV 2016. ISSN 0031-3203.
- JAKKULA, Vikramaditya. Tutorial on support vector machine (svm). **School of EECS, Washington State University**, v. 37, 2006.
- JIA, Weikuan; ZHAO, Dean; DING, Ling. An optimized RBF neural network algorithm based on partial least squares and genetic algorithm for classification of small sample. **APPLIED SOFT COMPUTING**, ELSEVIER SCIENCE BV, PO BOX 211, 1000 AE AMSTERDAM, NETHERLANDS, 48, p. 373–384, NOV 2016. ISSN 1568-4946.
- KAUFMAN, L.; ROUSSEEUW, Peter J. **Finding Groups in Data: An Introduction to Cluster Analysis**. [S.l.]: John Wiley, 1990.
- KECMAN, Vojislav; HUANG, Te-Ming.; VOGT, Michael. Iterative single data algorithm for training kernel machines from huge data sets: Theory and performance. In: _____. **Support Vector Machines: Theory and Applications**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. p. 255–274. ISBN 978-3-540-32384-6.
- KENNEDY, James. Particle swarm optimization. In: **Encyclopedia of machine learning**. [S.l.]: Springer, 2011. p. 760–766.
- KHARRAT, A.; HALIMA, M. B.; AYED, M. Ben. Mri brain tumor classification using support vector machines and meta-heuristic method. In: **ISDA, 15th International Conference on**. [S.l.: s.n.], 2015. p. 446–451.
- KOZA, John R. **Genetic programming: on the programming of computers by means of natural selection**. [S.l.]: MIT press, 1992.
- KOZA, John R. Genetic programming ii: Automatic discovery of reusable subprograms. **Cambridge, MA, USA**, 1994.
- LAARHOVEN, Peter JM Van; AARTS, Emile HL. Simulated annealing. In: **Simulated Annealing: Theory and Applications**. [S.l.]: Springer, 1987. p. 7–15.
- LINDEN, Ricardo. **Algoritmos genéticos (2a edição)**. Curitiba, Brasil: Brasport, 2008.
- MAVROFORAKIS, Michael E; THEODORIDIS, Sergios. A geometric approach to support vector machine (svm) classification. **IEEE transactions on neural networks**, IEEE, v. 17, n. 3, p. 671–682, 2006.
- MICHALEWICZ, Zbigniew. Gas: What are they? In: **Genetic Algorithms+ Data Structures= Evolution Programs**. [S.l.]: Springer, 1996. p. 13–31.
- MICHALEWICZ, Zbigniew; JANIKOW, Cezary Z. Handling constraints in genetic algorithms. In: **ICGA**. [S.l.: s.n.], 1991. p. 151–157.

- MIERSWA, Ingo. Evolutionary learning with kernels: a generic solution for large margin problems. In: **ACM. Proceedings of the 8th annual conference on Genetic and evolutionary computation**. [S.l.], 2006. p. 1553–1560.
- MITCHELL, Melanie. **"An Introduction to Genetic Algorithms"**. [S.l.]: "Paperback", 1998.
- MONTEIRO, S. T.; KOSUGI, Y. Applying particle swarm intelligence for feature selection of spectral imagery. In: **Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007)**. [S.l.: s.n.], 2007. p. 933–938. ISSN 2164-7143.
- MÜHLENBEIN, Heinz; SCHLIERKAMP-VOOSEN, Dirk. Predictive models for the breeder genetic algorithm i. continuous parameter optimization. **Evolutionary computation**, MIT Press, v. 1, n. 1, p. 25–49, 1993.
- NOCEDAL, Jorge; WRIGHT, Stephen J. **Sequential quadratic programming**. [S.l.]: Springer, 2006.
- OLIVEIRA, Saulo AF; BEZERRA, Francisco N; ROCHA NETO, Ajalmar R. Genetic seam carving: A genetic algorithm approach for content-aware image retargeting. In: **Pattern Recognition and Image Analysis**. Santiago de Compostela, Espanha: Springer, 2015. p. 700–707.
- OSUNA, Edgar; FREUND, Robert; GIROSI, Federico. Support vector machines: Training and applications. 1997.
- PLATT, J. C. Fast training of support vector machines using sequential minimal optimization. In: **Advances in Kernel Methods - SV Learning**. Cambridge, MA, USA: MIT Press, 1999.
- RECHENBERG, Ingo. Evolution strategy: Optimization of technical systems by means of biological evolution. **Fromman-Holzboog, Stuttgart**, v. 104, 1973.
- ROCHA NETO, Ajalmar R. **SINPATCO II: Novas Estratégias de Aprendizado de Máquinas para Classificação de Patologias da Coluna Vertebral**. Tese (Doutorado) — Tese (Doutorado)–Departamento de Engenharia de Teleinformática, Universidade Federal do Ceará, 2011.
- ROCHA NETO, Ajalmar R.; BARRETO, Guilherme A. Opposite maps: Vector quantization algorithms for building reduced-set svm and lssvm classifiers. **Neural Processing Letters**, v. 37, n. 1, p. 3–19, 2013.
- SILVA, Danilo Atilar; SILVA, Juliana Peixoto; ROCHA NETO, Ajalmar R. Novel approaches using evolutionary computation for sparse least square support vector machines. **Neurocomputing**, Elsevier, v. 168, p. 908–916, 2015.
- SIVANANDAM, S.N.; DEEPA, S.N. **Genetic Algorithms**. Heidelberg, Alemanha: Springer Berlin Heidelberg, 2008. 15–37 p. ISBN 978-3-540-73190-0. Disponível em: <http://dx.doi.org/10.1007/978-3-540-73190-0_2>.
- SIVAPATHASEKARAN, C; MUKHERJEE, Soumen; RAY, Arja; GUPTA, Ashish; SEN, Ramkrishna. Artificial neural network modeling and genetic algorithm based medium optimization for the improved production of marine biosurfactant. **Bioresource technology**, Elsevier, v. 101, n. 8, p. 2884–2887, 2010.
- SMOLA, AJ; SCHOLKOPF, B. A tutorial on support vector regression. **STATISTICS AND COMPUTING**, SPRINGER, VAN GODEWIJCKSTRAAT 30, 3311 GZ DORDRECHT, NETHERLANDS, 14, n. 3, p. 199–222, AUG 2004. ISSN 0960-3174.
- SMOLA, A J; SCHÖLKOPF, B. A tutorial on support vector regression. **Statistics and computing**, Springer, v. 14, n. 3, p. 199–222, 2004.

STOEAN, R.; DUMITRESCU, D.; PREUSS, M.; STOEAN, C. Evolutionary support vector regression machines. In: **2006 Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing**. [S.l.: s.n.], 2006. p. 330–335.

STOEAN, Ruxandra; PREUSS, Mike; STOEAN, Catalin; EL-DARZI, Elia; DUMITRESCU, D. Support vector machine learning with an evolutionary engine. **Journal of the Operational Research Society**, Nature Publishing Group, v. 60, n. 8, p. 1116–1122, 2009.

SYSWERDA, Gilbert. Uniform crossover in genetic algorithms. Morgan Kaufmann Publishers, Inc., 1989.

VAPNIK, Vladimir. **The Nature of Statistical Learning Theory**. [S.l.]: Springer, New York, 1995.

VAPNIK, V; GOLOWICH, SE; SMOLA, A. Support vector method for function approximation, regression estimation, and signal processing. In: Mozer, MC and Jordan, MI and Petsche, T (Ed.). **ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 9: PROCEEDINGS OF THE 1996 CONFERENCE**. FIVE CAMBRIDGE CENTER, CAMBRIDGE, MA 02142: M I T PRESS, 1997. (ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, 9), p. 281–287. ISBN 0-262-10065-7. ISSN 1049-5258. 10th Annual Conference on Neural Information Processing Systems (NIPS), DENVER, CO, DEC 02-05, 1996.

WANG, WJ; XU, ZB. A heuristic training for support vector regression. **Neurocomputing**, Elsevier, v. 61, p. 259–275, 2004.

WANG, WJ; XU, ZB; LU, WZ; ZHANG, XY. Determination of the spread parameter in the Gaussian kernel for classification and regression. **NEUROCOMPUTING**, ELSEVIER SCIENCE BV, PO BOX 211, 1000 AE AMSTERDAM, NETHERLANDS, 55, n. 3-4, p. 643–663, OCT 2003. ISSN 0925-2312.

ZHANG, Shike; LV, Jincheng; YUAN, Xinsheng; YIN, Shunde. BP Neural Network with Genetic Algorithm Optimization for Prediction of Geo-Stress State from Wellbore Pressures. **INTERNATIONAL JOURNAL OF COMPUTATIONAL INTELLIGENCE AND APPLICATIONS**, WORLD SCIENTIFIC PUBL CO PTE LTD, 5 TOH TUCK LINK, SINGAPORE 596224, SINGAPORE, 15, n. 3, SEP 2016. ISSN 1469-0268.