

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
PRÓ-REITORIA DE PESQUISA, PÓS-GRADUAÇÃO E INOVAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Pedro Hericson Machado Araújo

**Métodos para Obtenção de Modelos Reduzidos em Máquinas de
Vetores-Suporte por Mínimos Quadrados via Computação
Evolucionária**

Fortaleza – CE

13 de março de 2020

Pedro Hericson Machado Araújo

Métodos para Obtenção de Modelos Reduzidos em Máquinas de Vetores-Suporte por Mínimos Quadrados via Computação Evolucionária

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Linha de Pesquisa: Inteligência Artificial

Orientador: Prof. Dr. Ajalmar Rêgo da Rocha Neto

Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)

Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)

Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Fortaleza – CE

13 de março de 2020

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)

- A658m Araújo, Pedro Hericson Machado.
 Métodos para Obtenção de Modelos Reduzidos em Máquinas de Vetores-Suporte por Mínimos Quadrados via Computação Evolucionária / Pedro Hericson Machado Araújo. - 2020.
 109 f. : il. color.
- Dissertação (Mestrado) - Instituto Federal do Ceará, Mestrado em Ciência da Computação, Campus Fortaleza, 2020.
 Orientação: Prof. Dr. Ajalmar Rêgo da Rocha Neto.
1. Algoritmos Genéticos. 2. Métodos de Poda. 3. Máquinas de Vetores-Suporte por mínimos quadrados. 4. Seleção de Atributos e Padrões. I. Título.

CDD 004



Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE)
Pró-Reitoria de Pesquisa, Pós-Graduação e Inovação (PRPI)
Programa de Pós-Graduação em Ciência da Computação (PPGCC)

Pedro Hericson Machado Araújo

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

**Orientador: Prof. Dr. Ajalmar Rêgo da Rocha
Neto**

Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. Amauri Holanda de Souza Júnior
Instituto Federal de Educação, Ciência e Tecnologia do
Ceará (IFCE)

Prof. Dr. César Lincoln Cavalcante Mattos
Universidade Federal do Ceará (UFC)

Fortaleza – CE
13 de março de 2020

*Dedico este trabalho aos meus Pais
Pedro e Lucimar
Aos meus Amigos
e à minha querida namorada.*

Agradecimentos

Concluir um mestrado não é uma tarefa fácil. Nesse período de estudos, simulações e implementações tive apoio de pessoas importantes para a realização dessa fase.

Agradeço primeiramente à Deus e a todas as pessoas que entenderam minha ausência por conta do estudos, aos meus pais Pedro e Lucimar pelo total apoio, incentivo nos estudos e educação de qualidade durante a minha jornada, bem como tudo que fosse necessário para minha formação.

A Milena que me apoiou e incentivou durante a caminhada do mestrado.

Aos meus amigos pelos momentos de descontração, risadas e também pela ajuda em todos os quesitos, seja profissional, particular ou de ensino.

Ao Professor Ajalmar pela dedicação e orientação.

Aos membros do laboratório de inteligência artificial (LIA) pela ajuda nos momentos difíceis de implementação e que sempre estavam disponíveis para quaisquer dúvidas.

Ao programa de pós-graduação em ciência da computação (PPGCC) juntamente com os professores por compartilhar de forma brilhante seus conhecimentos. Ao Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE) Campus Maracanaú e Fortaleza, que estão presentes na minha vida há muitos anos, onde tive a honra de ser professor substituto e que considero a minha segunda casa.

*“Para nós os grandes homens não são aqueles que resolveram os problemas, mas aqueles
que os descobriram”.
(Albert Schweitzer)*

Resumo

Máquinas de Vetores-Suporte por Mínimos Quadrados (LSSVM) são uma alternativa à Máquina de Vetores-Suporte (SVM), com modificações na formulação do SVM, pois o processo de treinamento da LSSVM requer apenas a resolução de um sistema de equações lineares, ao invés de um problema de otimização de programação quadrática. Mas essas modificações resultaram na falta de uma característica importante presente na SVM: a detecção de forma automática de vetores-suporte, resultando ao classificador LSSVM no armazenamento de todos os padrões de treinamento como vetores-suporte ocasionando a falta de esparsidade, que pode ser prejudicial quando o conjunto de dados é demasiadamente grande.

Em alguns trabalhos presentes na literatura, Algoritmos Genéticos (AGs) e LSSVM trabalham juntos para encontrar parâmetros do classificador e/ou *kernel*. No entanto, nesta dissertação são propostos métodos que visam obter modelos esparsos combinando AGs e LSSVM para obtenção de modelos reduzidos. A imposição de esparsidade ao classificador LSSVM pode ser feita através de eliminação de atributos, bem como redução de vetores-suporte armazenados. Assim, a seleção de atributos tem como objetivo selecionar as características mais importantes, pois as que possuem baixa relevância e/ou redundância podem prejudicar o desempenho do classificador.

Diante disto, três novas abordagens foram propostas com o intuito de manter e até melhorar o desempenho obtido pelo classificador convencional. A proposta inicial utiliza algoritmos genéticos e tem como objetivo principal a seleção de atributos mais relevantes ao problema, visando sempre a obtenção da melhor acurácia. Levando em consideração que geralmente os conjuntos de dados possuem mais padrões que atributos, a seleção de características possui maior impacto que a remoção de padrões.

Na segunda proposta, a ideia é semelhante à primeira, com o diferencial em que a quantidade de atributos removidos é previamente definida, ou seja, o percentual de remoção possui tamanho fixo. Portanto, os operadores genéticos sofreram ajustes para adequação ao objetivo. A terceira proposta combina as vantagens da seleção de atributos em conjunto com a redução de vetores-suporte, trazendo uma abordagem dupla de redução, com a inclusão de um *score* que é utilizado para fins de comparação com os métodos já propostos na literatura. Foram realizadas simulações computacionais envolvendo conjuntos de dados do mundo real, comparando os métodos propostos com os presentes na literatura e assegurando o melhor desempenho obtido na maioria dos casos em termos de acurácia e redução quando comparados com outras versões do classificador LSSVM.

Palavras-chave: Métodos de Poda. Seleção de Atributos e Padrões. Máquina de Vetores-Suporte. Máquinas de Vetores-Suporte por Mínimos Quadrados. Algoritmos Genéticos.

Abstract

Least Square Support Vector Machines (LSSVM) are an alternative to the Support Vector Machine (SVM), with modifications to the SVM formulation, as the LSSVM training process requires only solving a system of linear equations, rather than of a quadratic programming optimization problem. But these modifications resulted in the lack of an important feature present in SVM: automatic detection of support vectors, resulting in the LSSVM classifier storing all training patterns as support vectors causing lack of sparsity, which can be detrimental when the data set is too large.

In some works in the literature, Genetic Algorithms (GAs) and LSSVM work together to find classifier parameters and/or kernel. However, this dissertation proposes methods that aim to get sparse models by combining AGs and LSSVM to obtain reduced models. The imposition of sparsity to the LSSVM classifier can be done by eliminating features as well as reducing stored support vectors. Thus, the features selection aims to select the most important features, since those with low relevance and/or redundancy may impair the performance of the classifier.

Given this, this work proposes three new approaches to maintain and even improve the performance obtained by the standard classifier. The initial proposal uses genetic algorithms and its main objective is to select the most relevant features to the problem, always aiming to get the best accuracy. Considering that datasets often have more patterns than features, the features selection has a greater impact than removing patterns.

In the second proposal, the idea is similar to the first, with the differential in which the amount of features removed is previously defined, that is, the percentage of removal is fixed beforehand. Therefore, genetic operators have undergone adjustments to fit the purpose. The third proposal combines the advantages of the features selection with support vector reduction, bringing a double reduction approach, with a score used for comparison to the methods already proposed in the literature.

Computational simulations were performed involving real-world datasets, comparing the proposed methods with those present in the literature and ensuring the best performance in most cases in terms of accuracy and reduction when compared to other versions of the LSSVM classifier.

Keywords: Pruning Methods. Features and Patterns Selection. Support Vector Machine. Least Squares Support Vector Machine. Genetic Algorithms

Lista de ilustrações

Figura 1 – Mapeamento não linear do espaço de dados de entrada para o espaço de características de alta dimensão.	36
Figura 2 – Demonstração das variáveis de folga da LSSVM.	37
Figura 3 – Processo de funcionamento do <i>K-fold</i>	43
Figura 4 – Fluxograma.	46
Figura 5 – Terminologia.	48
Figura 6 – Representações Genéticas.	49
Figura 7 – Seleção por Roleta.	50
Figura 8 – Seleção por <i>Rank</i>	50
Figura 9 – Seleção por Torneio.	51
Figura 10 – <i>Crossover</i> Uniponto.	52
Figura 11 – <i>Crossover</i> Dois Pontos.	52
Figura 12 – <i>Crossover</i> Uniforme.	53
Figura 13 – <i>Order Crossover</i>	54
Figura 14 – Mutação <i>bit-flip</i>	55
Figura 15 – Mutação por permutação.	55
Figura 16 – Indivíduo.	70
Figura 17 – Processo de transformação entre codificação binário-inteiro para aplicação do operador OX seguido pela transformação entre a codificação inteiro-binário.	74
Figura 18 – Sequência de passos realizados pela mutação.	75
Figura 19 – Indivíduo com dois cromossomos representando atributos (a) e padrões (b).	78
Figura 20 – Dois indivíduos realizando operações genéticas nos cromossomos equivalentes.	80
Figura 21 – Metodologia Utilizada para os métodos propostos	85
Figura 22 – Acurácia, número de componentes principais e percentual de informação presentes nos conjuntos de dados PID,VCP,BLD,HEP,SON e AUS.	88
Figura 23 – Acurácia e Poda para todos os conjuntos de dados.	91
Figura 24 – Acurácia e percentual de poda presentes nos conjuntos de dados.	94
Figura 25 – Gráficos de convergência para os conjuntos de dados.	107

Lista de tabelas

Tabela 1 – Listagem de <i>Kernels</i>	41
Tabela 2 – Amostra dos dados da coluna vertebral.	66
Tabela 3 – Amostra dos dados normalizados da coluna vertebral.	66
Tabela 4 – Matriz de covariância.	66
Tabela 5 – Autovetores da Coluna Vertebral.	67
Tabela 6 – Autovalores da Coluna Vertebral.	67
Tabela 7 – Percentual de Informação em cada componente principal.	67
Tabela 8 – Comparação de poda entre os classificadores.	77
Tabela 9 – <i>Datasets</i> utilizados.	83
Tabela 10 – Metodologia Utilizada.	85
Tabela 11 – Resultados para os classificadores LSSVM, PCA-LSSVM e FPGAS-LSSVM com 60%, 20% e 20% do conjunto de dados para treinamento, validação e teste, respectivamente.	87
Tabela 12 – Resultados para os classificadores FSFGAS-LSSVM, PCA-LSSVM e LSSVM com 60%, 20% e 20% do conjunto de dados completo para treinamento, validação e testes, respectivamente. <i>Acc</i> , θ e <i>#PR</i> são expressos em porcentagem, <i>#CT</i> em segundos.	90
Tabela 13 – Resultados para os classificadores LSSVM, PFGAS-LSSVM, P-LSSVM e PCA-LSSVM com 60%, 20% e 20% do conjunto de treinamento, validação e testes, respectivamente.	92
Tabela 14 – Resultados para os classificadores LSSVM, PFGAS-LSSVM, IP-LSSVM e PCA-LSSVM com 80% e 20% do conjunto de dados para treinamento e teste, respectivamente.	93

Lista de Algoritmos

1	Algoritmo Genético	47
2	FPGAS	71
3	FSFGAS	76
4	PFGAS	81

Lista de abreviaturas e siglas

RP	Reconhecimento de Padrões
SVM	<i>Support Vector Machine</i>
LSSVM	<i>Least Squares Support Vector Machine</i>
FPGAS-LSSVM	<i>Feature Pruning Genetic Algorithm for Sparse LSSVM</i>
FSFGAS-LSSVM	<i>Fixed Size Feature Genetic Algorithm for Sparse LSSVM</i>
PFGAS-LSSVM	<i>Patterns and Features Genetic Algorithm for Sparse LSSVM</i>
GAS-LSSVM	<i>Genetic Algorithm for Sparse LSSVM</i>
MOGAS-LSSVM	<i>Multi-Objective Genetic Algorithm for Sparse LSSVM</i>
FSGAS-LSSVM	<i>Fixed-Size Genetic Algorithm for Sparse LSSVM</i>
CE	Computação Evolucionária
AG	Algoritmo Genético
OX	<i>Order Crossover</i>
PCA	<i>Principal Component Analysis</i>
LDA	<i>Linear Discriminant Analysis</i>
KKT	<i>Karush-Kuhn-Tucker</i>
RBF	<i>Radial Basis Function</i>

Lista de símbolos

$\mathbf{x}$	Vetor de Atributos
y	Rótulo
$\mathbf{w}$	Vetor de Pesos do Hiperplano
ξ	Variável de Folga
γ	Termo de Regularização
b	bias
α	Vetor de multiplicadores de Lagrange
σ	Desvio Padrão
N_f	Número de Atributos
N_p	Número de Padrões de Treinamento
T_f	Total de atributos presentes após a remoção
N_{fr}	Número de Atributos Removidos
N_{pr}	Número de Padrões Removidos
CT	Tempo de Classificação em segundos
$P(t)$	População da Geração t
$\mathbf{Z}$	Conjunto de Dados
$\mathbf{Z}_1$	Conjunto de Treinamento
$\mathbf{Z}_2$	Conjunto de Validação
$\mathbf{Z}_3$	Conjunto de Teste
θ	Custo de Poda FSFGAS-LSSVM
Θ	Custo de Poda MOGAS-LSSVM
Q	Tamanho da População
$\phi(.)$	Mapeamento Realizado pela LSSVM

H_{+1}	Hiperplano da classe +1
H_{-1}	Hiperplano da classe -1
$\Re^n$	Conjunto de números reais n -dimensional
$\Re^{n_h}$	Conjunto dos números reais em um espaço de alta dimensão
$\mathbf{W}$	Matriz gerada pelo sistema de equações lineares da LSSVM
$\bar{x}$	Média do vetor $\mathbf{x}$
$\mathbf{e}$	Vetor de erro
λ_i	i -ésimo autovalor
$\mathbf{q}_i$	i -ésimo autovetor
$\mathbf{a}$	Vetor de componentes principais
$\mathbf{R}$	Matriz de Covariância

Sumário

1	INTRODUÇÃO	29
1.1	Objetivos Geral e Específicos	32
1.2	Produção científica	32
1.3	Estrutura da Dissertação	32
2	MÁQUINAS DE VETORES-SUORTE POR MÍNIMOS QUADRADOS - LSSVM	35
2.1	Introdução	35
2.2	Formulação Teórica	36
2.2.1	Formulação Primal	38
2.2.2	Vetores-Suporte	41
2.3	Obtenção dos Parâmetros Ótimos	42
2.3.1	Matriz Inversa	42
2.3.2	Matriz Pseudo-Inversa	42
2.3.3	Otimização de Hiperparâmetros	43
2.4	Conclusões	43
3	ALGORITMOS GENÉTICOS	45
3.1	Introdução	45
3.2	Fluxograma	46
3.3	Algoritmo	47
3.4	Terminologia	47
3.5	Função de Aptidão	48
3.6	Representação Genética	48
3.7	Métodos de Seleção	49
3.7.1	Seleção por Roleta	49
3.7.2	Seleção por <i>Rank</i>	50
3.7.3	Seleção por Torneio	51
3.7.4	Elitismo	51
3.8	Reprodução	51
3.8.1	<i>Crossover</i> Uniponto	51
3.8.2	<i>Crossover</i> dois pontos	52
3.8.3	<i>Crossover</i> Uniforme	52
3.8.4	<i>Order Crossover</i>	53
3.9	Mutação	54
3.9.1	<i>Bit Flip</i>	54

3.9.2	Mutação por Permutação (<i>Swap</i>)	55
3.10	Crítérios de Parada	55
3.11	Conclusão	56
4	OBTENÇÃO DE MODELOS ESPARSOS E REDUÇÃO DE DIMENSIONALIDADE	57
4.1	Métodos para Redução de Vetores-Suporte	57
4.1.1	<i>Opposite Maps</i>	57
4.1.2	LS ² -SVM	58
4.1.3	<i>Pruning</i> LSSVM	58
4.1.4	IP-LSSVM	59
4.2	Classificadores Esparsos Baseados em Algoritmos Genéticos	60
4.2.1	GAS-LSSVM	60
4.2.2	MOGAS-LSSVM	61
4.2.3	FSGAS-LSSVM	62
4.2.3.1	Reprodução	62
4.2.3.2	Mutação	62
4.3	Métodos para Seleção de Atributos	62
4.3.1	Análise de Componentes Principais	63
4.3.1.1	Redução de Dimensionalidade	64
4.3.1.2	PCA-LSSVM	67
4.3.2	Análise de Discriminantes Lineares	67
4.4	Conclusão	68
5	MÉTODOS PROPOSTOS	69
5.1	FPGAS-LSSVM (Proposta 1)	69
5.1.1	Indivíduos ou Cromossomos	69
5.1.2	Algoritmo	70
5.1.3	Função de <i>Fitness</i>	71
5.2	FSFGAS-LSSVM (Proposta 2)	71
5.2.1	Indivíduos ou Cromossomos	72
5.2.2	Função de <i>Fitness</i> para FSFGAS-LSSVM	73
5.2.3	Operador de Reprodução	73
5.2.4	Mutação	74
5.2.5	Algoritmo	75
5.3	PFGAS-LSSVM (Proposta 3)	76
5.3.1	Indivíduos ou Cromossomos	77
5.3.2	Função de <i>Fitness</i> para PFGAS-LSSVM	80
5.3.3	Reprodução	80
5.3.4	Mutação	80

5.3.5	Algoritmo	81
5.4	Conclusão	81
6	SIMULAÇÕES COMPUTACIONAIS	83
6.1	Metodologia	83
6.2	Resultados	86
6.2.1	Resultados para o classificador FPGAS-LSSVM (Proposta 1)	86
6.2.2	Resultados para o classificador FSFGAS-LSSVM (Proposta 2)	88
6.2.3	Resultados para o classificador PFGAS-LSSVM (Proposta 3)	89
6.3	Conclusão	95
7	CONCLUSÕES E TRABALHOS FUTUROS	97
7.1	Contribuições deste Trabalho	98
7.2	Trabalhos Futuros	98
	REFERÊNCIAS	99
	APÊNDICES	105
	APÊNDICE A – GRÁFICOS DE CONVERGÊNCIA	107

1 Introdução

Reconhecimento de padrões (RP) é uma área que tem como objetivo a classificação de objetos (padrões) em diversas categorias (classes). De acordo com as características do problema analisado, esses objetos podem representar imagens ou formas de ondas de áudio, bem como qualquer tipo de medição a ser classificada (THEODORIDIS; KOUTROUMBAS, 2009).

Até a década de 1960, o reconhecimento de padrões foi fortemente utilizado por pesquisadores na área de estatística. Entretanto, com o advento dos computadores, a demanda por aplicações práticas de reconhecimento de padrões foi intensificada, desencadeando novas demandas para desenvolvimentos teóricos. Assim, são várias as possíveis áreas de aplicação no âmbito de reconhecimento de padrões, tais como: reconhecimento de imagem, reconhecimento de áudio, biometria, bioinformática e mineração dos dados (THEODORIDIS; KOUTROUMBAS, 2009).

Aplicações em que os dados possuem rótulos são conhecidos como problemas de aprendizado supervisionado. Quando não se conhece as classes o problema é então denominado não supervisionado. Portanto, se para cada vetor de entrada existir finitas classes discretas, é possível dizer que se trata de um problema de classificação. Caso a saída desejada consistir em variáveis contínuas, então esse problema é de regressão (BISHOP, 2006).

Assim sendo, reconhecimento de padrões pode ser combinado com ferramentas no âmbito de Computação Evolucionária (CE) que abrange um conjunto de algoritmos com modo de atuação baseado na evolução natural (FOGEL; OWENS; WALSH, 1966; HOLLAND et al., 1992). Esses algoritmos correspondem a um conjunto de técnicas de otimização, que na maioria dos casos são inspiradas em processos naturais, como por exemplo a teoria de *Darwin*, com o intuito de encontrar soluções ótimas (IZZO; SPRAGUE; TAILOR, 2018).

Portanto, existe um princípio comum de funcionamento que está subjacente a todos os métodos da computação evolucionária: a população inicial, que contém um conjunto de possíveis soluções para um determinado problema. Tal população é constantemente modificada através da aplicação de operadores baseados na seleção natural, ou seja, evolui ao longo das gerações (PEREIRA, 2002).

Adotando os princípios de CE, os algoritmos genéticos (AGs) têm sido bastante utilizados como algoritmos adaptativos para a solução de problemas práticos e como modelos computacionais de sistemas evolutivos naturais em áreas do mundo real, tais como: bioinformática, medicina, economia, química e muitos outros (JAUHARI; RIZVI,

2014; FARAGÓ et al., 2014; ÁLVAREZ et al., 2012; PALENCIA; DELGADILLO, 2012).

Cientistas da computação trabalham em conjunto com biólogos para prever a estrutura das proteínas ou com cientistas políticos que modelam a evolução do comportamento cooperativo entre as nações (MITCHELL, 1998). Os AGs são procedimentos de busca probabilística projetados para trabalhar em grandes espaços de busca envolvendo estados que podem ser representados por *strings* (GOLDBERG; HOLLAND, 1988). Também são amplamente utilizados para maximizar a precisão dos classificadores usando métodos de poda (ALENCAR; ROCHA NETO; GOMES, 2016; BARBOSA; COSTA; ROCHA NETO, 2018; SILVA; ROCHA NETO, 2013; SILVA; SILVA; ROCHA NETO, 2015; SILVA; ROCHA NETO, 2015).

Máquina de Vetores-Suporte por Mínimos Quadrados (*Least Squares Support Vector Machine*, LSSVM) é uma teoria de aprendizagem proposta por Suykens e Vandewalle (1999a), que corresponde a uma versão modificada da Máquina de Vetores-Suporte (*Support Vector Machines*, SVM). O classificador LSSVM também pode ser utilizado para resolver problemas de regressão. Como as Máquinas de Vetores-Suporte (SVM), os classificadores LSSVM são aplicados para resolver problemas binários e podem, com algumas modificações, resolver problemas multiclasse. Em geral apresentam duas modificações importantes para os classificadores SVM. Ambas estão presentes na formulação do problema de otimização primal.

O classificador LSSVM é mais atraente em termos de computação, pois em geral, o custo computacional para resolução de equações lineares é menor em comparação com programação quadrática. Também envolve menos parâmetros de ajuste que a SVM convencional (SUYKENS; VANDEWALLE, 1999a), substituindo as restrições de desigualdade por restrições de igualdade no sentido de mínimos quadrados. No entanto, ao ter essa formulação, todas as amostras de entrada se tornam vetores-suporte, causando assim uma solução não esparsa.

Uma solução para a LSSVM é alcançada através da resolução de sistemas KKT Lineares¹ no sentido de mínimos quadrados. O classificador LSSVM tem sido amplamente utilizado na área de previsão de finanças (GESTEL et al., 2001), bem como previsão de carga elétrica (YUANCHENG; TINGJIAN; ERKENG, 2002). Infelizmente, a LSSVM não executa a seleção de modelos e perde a esparsidade de vetores-suporte, o que também prejudica o desempenho e generalização do SVM padrão (YANG et al., 2014).

Para lidar com a falta de esparsidade acima mencionada na LSSVM, vários métodos de conjuntos reduzidos (*reduced set*, RS) e poda foram propostos (SILVA; SILVA; ROCHA NETO, 2015; SILVA; ROCHA NETO, 2015; CARVALHO; BRAGA, 2009). Esses métodos compreendem várias técnicas que visam simplificar a estrutura interna desses modelos,

¹ Karush-Kuhn-Tucker systems.

mantendo os limites de decisão o mais semelhante possível aos originais. Eles são muito úteis na redução de complexidade computacional dos modelos originais, pois agilizam o processo de decisão, reduzindo o número de vetores-suporte. Sendo particularmente importantes para lidar com grandes conjuntos de dados quando um grande número de amostras de dados podem ser selecionadas como vetores-suporte, de modo que é possível a remoção de vetores-suporte menos relevantes (GEEBELEN; SUYKENS; VANDEWALLE, 2012) ou construindo um conjunto menor de exemplos de treinamento (PERES; PEDREIRA, 2009).

Vale salientar que a remoção de padrões é uma maneira de obter modelos esparsos. No entanto, também é possível obter modelos LSSVM esparsos, eliminando atributos do conjunto de dados. Dessa forma, cada atributo removido evita que N_p valores sejam salvos, onde N_p é o número de padrões no conjunto de dados de treinamento. Além disso, a remoção de um padrão tem, em geral, menos impacto do que a remoção de uma característica, pois o conjunto de dados de treinamento normalmente possui mais padrões que atributos.

Diante disto, para combinar as vantagens acima mencionadas do classificador LSSVM e algoritmos genéticos, esta dissertação foca em modelos reduzidos com o intuito de seleção de atributos e padrões. De modo que foram propostos três métodos como alternativas às metodologias existentes para alcançar modelos esparsos, sendo abordados nas Seções 5.1, 5.2 e 5.3. Os métodos propostos são baseados em classificadores esparsos com auxílio de AGs. Diferentemente de trabalhos anteriores presentes na literatura, meta-heurísticas são utilizadas para obter modelos reduzidos e impor esparsidade, não para descobrir valores ótimos dos parâmetros de *kernel* e/ou do classificador (ARAÚJO; ROCHA NETO, 2019).

Assim, a seleção de atributos representa um problema de fundamental importância em aprendizado de máquina, sendo constantemente utilizada como uma etapa de pré-processamento de dados. A ideia é selecionar os atributos mais importantes, visto que os menos relevantes e/ou redundantes podem prejudicar a precisão do classificador. Com o intuito de selecionar ou descartar atributos, é necessária uma forma de determinar a importância de cada atributo do conjunto de dados (LEE, 2005).

Seguindo essa ideia, a análise de componentes principais (*Principal Component Analysis*, PCA) pode ser utilizada para reduzir a dimensionalidade do espaço de atributos, conseguindo determinar a quantidade de informação presente em cada característica.

1.1 Objetivos Geral e Específicos

Esta dissertação tem como objetivo realizar um estudo comparativo entre métodos de poda pertinentes ao classificador LSSVM com o intuito de obter modelos reduzidos através de seleção de atributos e vetores-suporte. Adicionalmente, almeja-se avaliar o uso de meta-heurística para obtenção de modelos esparsos.

Em seu sentido mais particular, os seguintes objetivos específicos são:

- Apresentar novas propostas para obtenção de modelos reduzidos em LSSVM para tarefas de classificação;
- Apresentar dois classificadores esparsos de seleção de atributos com base em AGs;
- Apresentar um novo classificador esparsos de seleção de padrões e atributos utilizando AGs;
- Comparar os resultados obtidos a partir das propostas com outros métodos presentes na literatura;

1.2 Produção científica

Durante este projeto de mestrado, o seguinte trabalho científico foi aceito e publicado, a saber:

- **Araújo, P.H.M.; ROCHA NETO, A.R.. Sparse Least Squares Support Vector Machines Based on Genetic Algorithms: A Feature Selection Approach.** In: International Work-Conference on Artificial Neural Networks - IWANN 2019. Gran Canaria, Spain: Springer International Publishing, 2019, (Lecture Notes in Computer Science, v. 11507). p. 500-511. Disponível em <https://link.springer.com/chapter/10.1007/978-3-030-20518-8_42>

1.3 Estrutura da Dissertação

Esta dissertação possui 7 Capítulos e os remanescentes são: Máquinas de Vetores-Suporte por Mínimos Quadrados - LSSVM, Algoritmos Genéticos -AG, Obtenção de Modelos Esparsos e Redução de Dimensionalidade, Métodos Propostos, Simulações Computacionais e por fim, Conclusões e Trabalhos Futuros.

A ideia durante os capítulos é abordar o conteúdo de forma clara e intuitiva, onde o leitor compreenda os objetivos desta dissertação, bem como apresentações dos conteúdos existentes na literatura.

O Capítulo 2 aborda os fundamentos teóricos do classificador Máquina de Vetores-Suporte por Mínimos Quadrados (LSSVM), bem como o problema da falta de esparsidade, objeto de estudo dessa dissertação. De forma sucinta são descritas as diferenças com o classificador Máquina de Vetores-Suporte (SVM).

Como os métodos propostos por esta dissertação utilizam de algoritmos genéticos (AGs) e LSSVM, o Capítulo 3 aborda o contexto em que se apresenta os AGs para melhor compreensão deste método, bem como modelos de seleção, operadores tradicionais de mutação e reprodução, exemplificando a evolução das gerações.

O Capítulo 4 exemplifica propostas presentes na literatura para seleção de vetores-suporte e redução de dimensionalidade, algumas delas foram utilizadas para fins de comparação, já outras foram abordadas de forma abreviada.

O Capítulo 5 possui as mais expressivas contribuições desta dissertação. Os métodos propostos *Feature Pruning Genetic Algorithm for Sparse LSSVM* (FPGAS-LSSVM) e *Fixed Size Feature Genetic Algorithm for Sparse LSSVM* (FSFGAS-LSSVM) são exemplificados, tendo como base a seleção de atributos do conjunto de dados com algoritmos genéticos para impor esparsidade ao classificador bem como manter ou até mesmo melhorar a acurácia.

Basicamente a diferença entre os dois métodos acima mencionados é referente à quantidade de atributos removidos. O método FSFGAS-LSSVM realiza as operações com uma quantidade fixa de características removidas que devem ser informadas previamente por um parâmetro θ . O método FPGAS-LSSVM obtém a taxa de poda que melhor maximiza a acurácia de forma automática. O último método proposto combina as vantagens da seleção de atributos e remoção de vetores-suporte denominado *Patterns and Features Genetic Algorithm for Sparse LSSVM* (PFGAS-LSSVM) para obter a melhor acurácia e taxa de poda.

O Capítulo 6 demonstra as simulações computacionais realizadas para obtenção dos resultados, bem como tabelas e gráficos. Também ilustra a metodologia abordada e especificações das bases de dados utilizadas.

Por fim, o Capítulo 7 descreve sobre as conclusões obtidas, bem como ideias e evoluções em trabalhos futuros.

2 Máquinas de Vetores-Suporte por Mínimos Quadrados - LSSVM

Neste capítulo é apresentada a LSSVM que pode ser facilmente implementada devido a algumas modificações com relação à formulação da SVM que foi concebida a partir da teoria do aprendizado estatístico apresentada por Vapnik (1995). Uma das principais diferenças é dada pelo fato de que todos os dados do conjunto de treinamento são considerados vetores-suporte. As Seções 5.1, 5.2 e 5.3 exemplificam três métodos para minimizar esse problema, ou seja, obter classificadores esparsos.

2.1 Introdução

As máquinas de vetores-suporte não-lineares possuem uma boa característica de resolver problemas não-lineares de classificação e regressão através de programação quadrática. Com o intuito de facilitar o esforço computacional sem perder a qualidade nas soluções obtidas, Suykens e Vandewalle (1999a) introduziram modificações nas máquinas de vetores-suporte que foi denominada de Máquina de Vetores-Suporte por Mínimos Quadrados (*Least Squares Support Vector Machines*, LSSVM), podendo também ser utilizada em problemas de classificação e regressão. Com relação ao quesito classificação¹, sua formulação é baseada em problemas binários, podendo ser estendido para abordar problemas multiclasse (SUYKENS; VANDEWALLE, 1999b; GESTEL et al., 2002).

Existem diversas aplicações do uso do classificador LSSVM tais como em tarefas de:

- Classificação
 - Reconhecimento de dígitos
 - Detecção e identificação de faces
 - Classificação de textos
- Regressão
 - Predição de séries temporais
 - Previsão de índices de bolsa de valores

Nessa nova abordagem, existe a utilização de restrições de igualdade no problema primal a ser minimizado, em contraste com o SVM que utiliza desigualdade. De modo que seja possível agora resolver problemas por meio de equações lineares, diferente da SVM que utiliza programação quadrática.

¹ Modalidade abordada por esta dissertação.

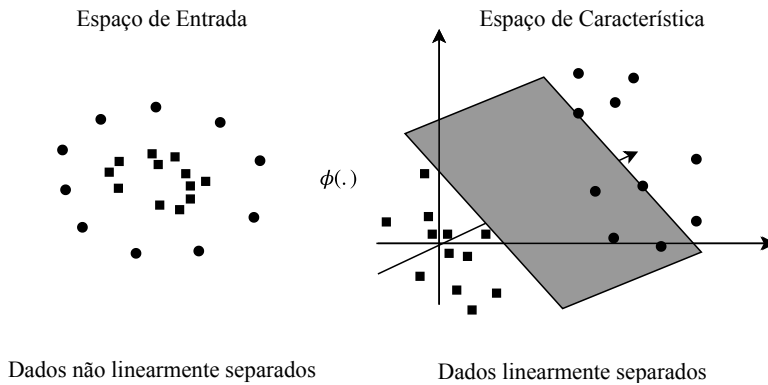
2.2 Formulação Teórica

A partir de um conjunto de treinamento $\{\mathbf{x}_i, y_i\}_{i=1}^{N_p}$, com os padrões $\mathbf{x}_i \in \mathbb{R}^n$ e rótulos binários $y_i \in \{-1, +1\}$, a ideia é realizar o mapeamento dos dados no espaço de características de dimensão mais elevada, conseguindo assim construir uma superfície linear de separação $f(x) = 0$, tal que:

$$\mathbf{w}^T \phi(\mathbf{x}) + b = 0 \quad (2.1)$$

Onde $\mathbf{w} \in \mathbb{R}^n$ representa o vetor de pesos, b é o *bias* e $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^{n_h}$ é o mapeamento realizado em alta dimensão no espaço de características (Figura 1), que pode ser dimensão infinita, ou seja, $n_h = \infty$ (OLIVEIRA; GOMES; ROCHA NETO, 2018).

Figura 1 – Mapeamento não linear do espaço de dados de entrada para o espaço de características de alta dimensão.



Fonte: Próprio Autor.

A LSSVM possui o espaço de características composto por dois hiperplanos paralelos, onde cada um representa suas respectivas classes como mostra a Figura 2 derivada de Carvalho e Braga (2009). A correta classificação do padrão $\mathbf{x}_i$ sendo que $y_i = +1$ se dá quanto menor for a distância com relação ao hiperplano H_{+1} . De forma semelhante quando o padrão $\mathbf{x}_i$ tal que $y_i = -1$ ocorre quanto menor sua distância em relação ao hiperplano H_{-1} .

Sendo assim, o espaço de características do classificador LSSVM passa a ser dividido em dois hiperplanos:

$$\begin{cases} H_{+1} : \mathbf{w}^T \phi(\mathbf{x}_i) + b = +1 \text{ se } y_i = +1 \\ H_{-1} : \mathbf{w}^T \phi(\mathbf{x}_i) + b = -1 \text{ se } y_i = -1 \end{cases} \quad (2.2)$$

Recombinando a Eq. (2.2), tem-se:

$$y_i \{ [\mathbf{w}^T \phi(\mathbf{x}_i)] + b \} = 1, i = 1 \dots N_p \quad (2.3)$$

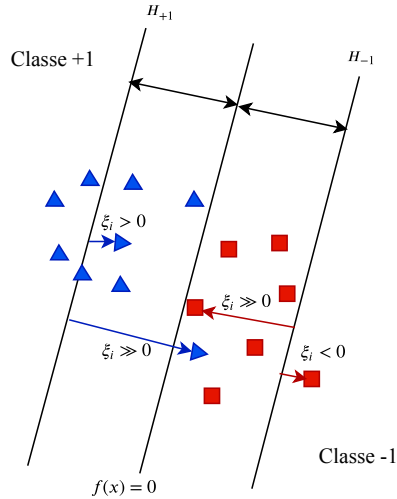
Uma variável de folga ξ_i é permitida, de modo que erros de classificação sejam tolerados, ou seja, permite que alguns padrões de treinamento sejam incorretamente classificados. Principalmente em casos de distribuições sobrepostas.

Com a inclusão da variável de folga ξ_i , é obtida a nova equação, como segue:

$$y_i \{ [\mathbf{w}^T \phi(\mathbf{x}_i)] + b \} = 1 - \xi_i, i = 1 \dots N_p \quad (2.4)$$

Contudo, a variável de folga ξ dos classificadores LSSVM pode assumir qualquer valor real, diferente da SVM, onde sua variável de folga é sempre positiva. A variável ξ indica a distância do padrão $\mathbf{x}_i$ em relação ao hiperplano referente a sua classe como mostra a Figura 2 derivada de Carvalho e Braga (2009). Portanto, é possível determinar se um vetor está próximo ou não da superfície de separação.

Figura 2 – Demonstração das variáveis de folga da LSSVM.



Fonte: Próprio Autor.

Quando a variável de folga possui valores negativos, $\xi_i < 0$, significa que o padrão $\mathbf{x}_i$ está distante da superfície de separação, sua classificação é correta. Portanto, quanto menor o valor de ξ_i , mais distante dos hiperplanos H_{+1} e H_{-1} se encontra o padrão $\mathbf{x}_i$. Já para o valor da variável de folga $\xi_i = 0$, tem-se que o vetor encontra-se exatamente no hiperplano relacionado a sua classe, sendo assim caracteriza uma classificação correta do padrão, pois os hiperplanos tendem a ficar afastados, vale salientar que este é um caso raro de ocorrer.

Para valores positivos, $\xi_i > 0$ significa que o vetor se encontra entre os dois hiperplanos paralelos, próximo ao hiperplano de sua classe. Portanto, quanto maior o valor de ξ_i , mais próximo do hiperplano incorreto ele estará e conseqüentemente mais longe estará do hiperplano correto, ou seja, $\xi_i \gg 0$ indicam padrões localizados na região de fronteira ou mesmo na região de outra classe (classificados incorretamente). Levando em consideração que a análise de apenas o valor de ξ_i dificulta a determinação objetiva da classificação do padrão, a única afirmação que pode ser feita nesta situação é que quanto maior o valor de ξ_i , mais longe do hiperplano correto o padrão se encontra.

2.2.1 Formulação Primal

Semelhante aos classificadores SVM, o objetivo do processo de treinamento da LSSVM é obter os valores dos pesos $\mathbf{w}$ e para o b (*bias*), partindo de um problema primal dado por:

$$\begin{aligned} \min J_p(\mathbf{w}, b, \boldsymbol{\xi}) &= \left\{ \frac{1}{2} \mathbf{w}^T \mathbf{w} + \gamma \frac{1}{2} \sum_{i=1}^{N_p} \xi_i^2 \right\} \\ \text{sujeito a } y_i \{ [\mathbf{w}^T \phi(\mathbf{x}_i)] + b \} &= 1 - \xi_i, i = 1, \dots, N_p \end{aligned} \quad (2.5)$$

Analisando a função custo (2.5), o primeiro termo tem como objetivo a maximização da distância entre os dois hiperplanos paralelos. O segundo termo da função custo tem a finalidade de minimizar as variáveis de folga ξ_i com o intuito de garantir que os hiperplanos paralelos estejam em uma localização aproximadamente central em relação aos vetores de cada classe (CARVALHO, 2005). É interessante salientar que a função de custo presente na LSSVM abrange a soma de dois termos quadráticos, motivo pelo qual a minimização é denominada de mínimos quadrados.

Algumas mudanças foram necessárias em relação ao classificador SVM, a primeira é na utilização de restrições de igualdade ao invés de desigualdade. A segunda mudança é referente a função de custo, onde as variáveis de folga ξ_i indicam a distância do vetor $\mathbf{x}_i$ ao seu hiperplano correspondente e são representadas a partir da incorporação da soma das variáveis ao quadrado, juntamente com um termo de regularização γ . Aplicando-se o método dos multiplicadores de Lagrange (BERTSEKAS; RHEINBOLDT, 1982), com o intuito de incorporar na função de custo dual às restrições de igualdade do problema primal, obtém-se a seguinte função lagrangeana para classificadores LSSVM:

$$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \gamma \frac{1}{2} \sum_{i=1}^{N_p} \xi_i^2 - \sum_{i=1}^{N_p} \alpha_i \{ y_i [\mathbf{w}^T \phi(\mathbf{x}_i) + b] - 1 + \xi_i \}, \quad (2.6)$$

Onde γ é um parâmetro de custo positivo, $\{\xi_i\}_{i=1}^{N_p}$ são as variáveis de folga, b é o *bias* e $\{\alpha_i\}_{i=1}^{N_p}$ são os multiplicadores de Lagrange quase sempre não nulos. A formulação do SVM proposta por Vapnik (1995) é modificada por Suykens e Vandewalle (1999a) em dois pontos. No primeiro ponto, ao invés de restrições de desigualdade², utiliza restrições de igualdade onde o valor 1 do lado direito pode ser considerado como um valor alvo ao invés de um valor limite.

A segunda mudança refere-se à incorporação na função custo da soma das variáveis de folga ao quadrado, $\sum_{i=1}^{N_p} \xi_i^2$, ponderada por uma constante de regularização γ que tem o mesmo significado da constante C apresentada no SVM (SILVA, 2015). A formulação da LSSVM permite resolver um problema de otimização de forma mais simples, necessitando apenas de um sistema de equações lineares, mais precisamente de um sistema denominado *Karush-Khun-Tucker* (KKT) (FLETCHER, 2013).

Vale salientar que o classificador LSSVM possui algumas limitações, uma delas é a falta de esparsidade³ na solução dos problemas, em outras palavras, os multiplicadores de Lagrange associados α são não nulos⁴. Após o processo de treinamento, todos os vetores-suporte precisam ser armazenados, assim como os multiplicadores de Lagrange associados para realizar a etapa de classificação do algoritmo, o que pode não ser adequado para conjunto de dados demasiadamente grandes.

As condições para otimalidade são dadas pelas derivadas parciais de forma semelhante ao classificador SVM.

$$\begin{aligned}
 \frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha})}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N_p} \alpha_i y_i \phi(\mathbf{x}_i), \\
 \frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha})}{\partial b} &= - \sum_{i=1}^{N_p} \alpha_i y_i, \\
 \frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha})}{\partial \alpha_i} &= -y_i [\mathbf{w}^T \phi(\mathbf{x}_i) + b] + 1 - \xi_i, \\
 \frac{\partial L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha})}{\partial \xi_i} &= \gamma \sum_{i=1}^{N_p} \xi_i - \sum_{i=1}^{N_p} \alpha_i.
 \end{aligned} \tag{2.7}$$

De acordo com a Eq. (2.7) é possível notar que o valor de α pode assumir qualquer valor real, pois está associado a uma restrição de igualdade, o que difere da SVM. Portanto, a solução para o problema *primal* de otimização para a função lagrangeana se dá pela resolução das derivadas parciais presentes na Eq. (2.7). As condições de otimalidade presentes na Eq. (2.7) podem ser igualadas a zero, com o intuito de encontrar o ponto de sela da função lagrangeana.

² Utilizada nos classificadores SVM.

³ Problema abordado como tema desta dissertação.

⁴ Pode acontecer de algum multiplicador de Lagrange ser nulo.

Assim, com base na Eq. (2.7), é possível formular um sistema de equações lineares denominado *Karush-Kuhn-Tucker*(KKT) para representar esse problema como:

$$\begin{cases} \mathbf{w} = \sum_{i=1}^{N_p} \alpha_i y_i \phi(\mathbf{x}_i), \\ \sum_{i=1}^{N_p} \alpha_i y_i = 0, \\ y_i [\mathbf{w}^T \phi(\mathbf{x}_i) + b] - 1 + \xi_i = 0, \forall i, \\ \alpha_i = \gamma \xi_i, \forall i. \end{cases} \quad (2.8)$$

A falta de esparsidade é uma desvantagem dessa abordagem, a condição de otimalidade $\alpha_i = \gamma \xi_i$ deixa isto bem claro, onde na LSSVM cada padrão de treinamento é considerado um vetor de suporte. Como na maioria dos casos nenhum valor de α_i será nulo, pois ele é proporcional à variável de folga ξ_i que dificilmente terá valor zero, o algoritmo sofre com a falta de esparsidade. Portanto, todos os dados do treinamento irão contribuir para o modelo (SUYKENS; GESTEL; BRABANTER, 2002). Com o intuito de obter a superfície de decisão, os termos $\mathbf{w}$ e $\phi(\mathbf{x}_i)$ não são calculados de forma direta (SUYKENS; VANDEWALLE, 1999a), de modo que o sistema KKT pode ser resolvido como se segue:

$$\mathbf{W}\mathbf{z} = \mathbf{1}, \quad (2.9)$$

onde

$$\mathbf{W} = \left[\begin{array}{c|c} 0 & \mathbf{y}^T \\ \hline \mathbf{y} & \Omega + \gamma^{-1}\mathbf{I} \end{array} \right], \mathbf{z} = \left[\begin{array}{c} b \\ \boldsymbol{\alpha} \end{array} \right], \mathbf{1} = \left[\begin{array}{c} 0 \\ \mathbf{1} \end{array} \right], \quad (2.10)$$

$\Omega \in \Re^{N_p \times N_p}$ é a matriz onde as entradas são dadas por $\Omega_{i,j} = y_i y_j \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, $i, j = 1, \dots, N_p$, $y = [y_1 \dots y_{N_p}]^T$ e o símbolo $\mathbf{1}$ denota um vetor de uns com dimensão N_p .

A solução deste sistema linear pode ser calculada pela inversão direta da matriz $\mathbf{W}$ como segue⁵:

$$\mathbf{z} = \mathbf{W}^{-1}\mathbf{1} = \left[\begin{array}{c|c} 0 & \mathbf{y}^T \\ \hline \mathbf{y} & \Omega + \gamma^{-1}\mathbf{I} \end{array} \right]^{-1} \left[\begin{array}{c} 0 \\ \mathbf{1} \end{array} \right], \quad (2.11)$$

Uma vez obtidos os valores dos multiplicadores de Lagrange e o *bias*, de modo que utilizando o truque do *kernel* é possível substituir $\Omega_{i,j} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$, a saída pode ser calculada com base na função de classificação descrita como:

⁵ A matriz $\mathbf{W}$ gerada pelo sistema de equações lineares da LSSVM possui tamanho $(N_p + 1) \times (N_p + 1)$.

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i=1}^{N_p} \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i) + b \right). \quad (2.12)$$

É fácil ver na Eq. (2.12) que sempre que um multiplicador de Lagrange α_i é zero, não é necessário manter o vetor de entrada associado $\mathbf{x}_i$ para uso futuro. Também é importante o uso do truque do *kernel*, que é aplicado para gerar versões não lineares do classificador SVM linear padrão. Este procedimento funciona substituindo o produto $\mathbf{x}^T \mathbf{x}_i$ com a função de *kernel* $K(\mathbf{x}, \mathbf{x}_i)$. A Tabela 1 ilustra alguns exemplos de *kernel*.

Tabela 1 – Listagem de *Kernels*

<i>Kernel</i>	Descrição	Detalhes
Linear	$K(\mathbf{x}, \mathbf{x}_i) = \mathbf{x}^T \mathbf{x}_i$	
Polinomial	$K(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^T \mathbf{x}_i + 1)^d$	d é um parâmetro
RBF	$K(\mathbf{x}, \mathbf{x}_i) = \exp\{-\ \mathbf{x} - \mathbf{x}_i\ ^2 / \sigma^2\}$	σ é um parâmetro

Fonte: Próprio Autor.

2.2.2 Vetores-Suporte

A grande desvantagem da LSSVM é relacionada ao fato de que todos os padrões do treinamento são armazenados e selecionados como vetores-suporte, portanto, para conjuntos de dados demasiadamente grandes, o treinamento pode se tornar inviável. Tendo em vista que ξ_i apenas terá valor zero quando o vetor suporte associado estiver localizado no espaço de características, exatamente sobre um ponto que corresponde a sua classe, o que dificilmente ocorre. Portanto, esta dissertação tem como foco impor esparsidade ao modelo, removendo atributos/padrões irrelevantes para o problema, de modo que mantém o desempenho do classificador e algumas vezes consegue melhorá-lo.

A seleção automática de vetores-suporte possui vantagens como pode ser visto a seguir (LEE; MANGASARIAN, 2001a):

- Com o menor número de vetores de treinamento armazenados, gasta-se menos memória.
- Custo computacional para avaliar um novo padrão é reduzido.
- Melhor análise dos vetores-suporte mais relevantes para o problema, quando um número menor é obtido.
- Possibilidade de representação de uma base de dados grande por meio de poucos vetores.
- Aplicações que necessitam de treinamentos contínuos podem ser executadas sem a preocupação com acúmulo de dados.

2.3 Obtenção dos Parâmetros Ótimos

2.3.1 Matriz Inversa

Para a obtenção dos termos α e b presentes na solução do sistema linear apresentado na Eq. (2.9), é necessário apenas a inversão da matriz $\mathbf{W}$, resolvendo o sistema

$$\begin{aligned}\mathbf{W}\mathbf{z} &= \mathbf{1} \\ \mathbf{W}^{-1}\mathbf{W}\mathbf{z} &= \mathbf{W}^{-1}\mathbf{1} \\ \mathbf{z} &= \mathbf{W}^{-1}\mathbf{1}\end{aligned}\tag{2.13}$$

ou seja

$$\mathbf{z} = \left[\begin{array}{c|c} 0 & \mathbf{y}^T \\ \hline \mathbf{y} & \Omega + \gamma^{-1}\mathbf{I} \end{array} \right]^{-1} \left[\begin{array}{c} 0 \\ \mathbf{1} \end{array} \right],\tag{2.14}$$

necessitando apenas que a matriz $\mathbf{W}$ seja não-singular.

2.3.2 Matriz Pseudo-Inversa

O método de solução mais simples, utiliza o cálculo da inversa da matriz $\mathbf{W}$, mas quando se obtém uma matriz não-quadrada é necessário resolver os problemas com base na pseudo-inversa (MOORE, 1920; PENROSE, 1955). Em geral, somente as colunas da matriz $\mathbf{W}$ serão removidas, o que representa valores dos multiplicadores de Lagrange associados a determinadas colunas serem nulos (LEE; MANGASARIAN, 2001b). Entretanto, a remoção das linhas equivale à remoção das restrições associadas ao problema de otimização. Para obter o vetor solução $\mathbf{z}$, para uma matriz não quadrada $\mathbf{W}_1$ é necessário realizar:

$$\begin{aligned}\mathbf{W}_1\mathbf{z} &= \mathbf{1} \\ (\mathbf{W}_1^T\mathbf{W}_1)\mathbf{z} &= \mathbf{W}_1^T\mathbf{1} \\ (\mathbf{W}_1^T\mathbf{W}_1)^{-1}(\mathbf{W}_1^T\mathbf{W}_1)\mathbf{z} &= (\mathbf{W}_1^T\mathbf{W}_1)^{-1}\mathbf{W}_1^T\mathbf{1} \\ \mathbf{I}\mathbf{z} &= (\mathbf{W}_1^T\mathbf{W}_1)^{-1}\mathbf{W}_1^T\mathbf{1} \\ \mathbf{z} &= (\mathbf{W}_1^T\mathbf{W}_1)^{-1}\mathbf{W}_1^T\mathbf{1}\end{aligned}\tag{2.15}$$

Logo, tem-se que

$$\mathbf{z} = \mathbf{W}^\dagger\mathbf{1}\tag{2.16}$$

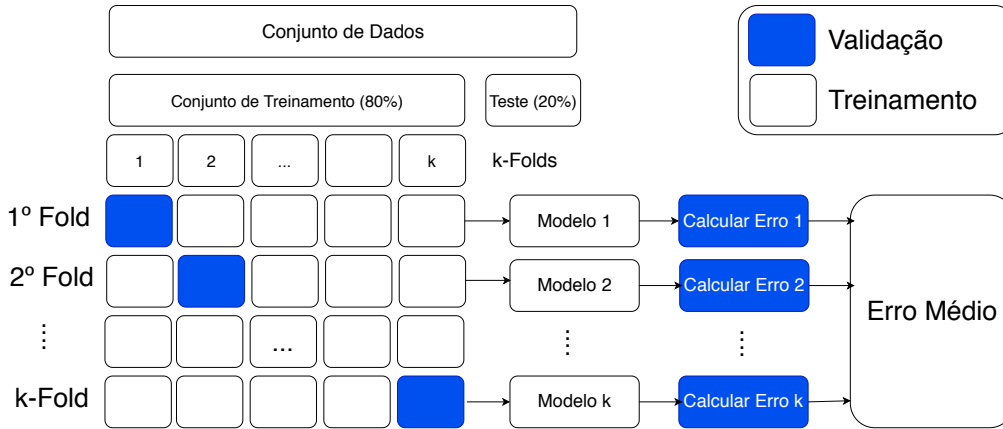
onde

$$\mathbf{W}^\dagger = (\mathbf{W}_1^T \mathbf{W}_1)^{-1} \mathbf{W}_1^T. \quad (2.17)$$

2.3.3 Otimização de Hiperparâmetros

Para a escolha de hiperparâmetros, como, por exemplo o termo de regularização γ é recomendado a utilização de métodos como busca em grade (*grid search*) em conjunto com validação cruzada, onde o *k-fold* é o mais conhecido. A ideia é dividir o conjunto de dados de treinamento em k subconjuntos de mesmo tamanho, onde se faz necessário a utilização de $k - 1$ subconjuntos para treino e a parte remanescente para validação. O processo é realizado k vezes circularmente. Por fim, calcula-se o erro médio para todos os k modelos. A Figura 3 demonstra o funcionamento.

Figura 3 – Processo de funcionamento do *K-fold*.



Fonte: Próprio Autor.

2.4 Conclusões

Neste capítulo foram apresentadas as características teóricas do classificador LS-SVM, bem como facilidade da implementação por meio de sistemas lineares e construção de hiperplanos paralelos para a maximização da margem de separação. Também foram apresentadas desvantagens em relação à falta de esparsidade, ou seja, a falta da detecção automática de vetores-suporte da solução. Isto resulta no armazenamento de todos os padrões de treinamento, dificultando o processo para bases demasiadamente grandes. Em casos em que a matriz $\mathbf{W}$ seja não quadrada, o método da pseudo-inversa (2.15) pode ser utilizado, já para a obtenção dos parâmetros de *kernel* e termo de regularização γ , existe o conceito de utilização de métodos como busca em grade em conjunto com validação cruzada.

3 Algoritmos Genéticos

Neste capítulo serão apresentados os conceitos iniciais que envolvem os algoritmos genéticos, bem como o processo de evolução baseado na teoria de *Darwin*. O intuito deste capítulo é descrever o funcionamento dessa meta-heurística para melhor entendimento dos métodos propostos (Capítulo 5) para obtenção de modelos reduzidos, bem como o funcionamento dos operadores tradicionais de seleção, reprodução e mutação.

3.1 Introdução

Algoritmo Genético (AG) é um método de busca meta-heurístico inspirado por evolução natural, como herança, mutação, seleção natural e *crossover*. Essa meta-heurística pode ser usada para gerar soluções úteis para problemas de busca e otimização. Em algoritmos genéticos, uma população contém indivíduos candidatos para melhor resolver o problema de otimização envolvendo a melhor solução através da seleção natural, ou seja, a função de adequação.

Devido às características subjacentes do AG, é possível resolver alguns problemas por esse método sem a necessidade de suposição de linearidade, continuidade, diferenciabilidade ou convexidade da função objetivo, diferentemente do que ocorre com outros modelos matemáticos. Cada indivíduo da população possui genes (vetor de genes) chamados cromossomos, para que possam ser modificados por operações de mutação e combinados com outros indivíduos para gerar novos indivíduos (reprodução) usando os operadores de cruzamento. Neste trabalho o vetor de genes tem representação binária com string de 0s e 1s, outros modelos de representação são possíveis.

A população inicial contém indivíduos gerados aleatoriamente, que são chamados de geração. O valor da função de adequação indica quão boa é a solução. Esse valor é utilizado para guiar o processo de reprodução, no qual indivíduos com alto valor de aptidão são mais propensos a distribuir seus genes na nova população a serem obtidos através de um processo de reprodução. O valor de avaliação é calculado após a decodificação do cromossomo (indivíduo) e a melhor solução para o problema a ser resolvido é, portanto, aquele com o maior valor para a função de aptidão. Para determinar o melhor indivíduo da população, calcula-se a precisão ou taxa de acerto de cada indivíduo, construindo uma classificação de quão boa é a solução. Os melhores indivíduos permanecem nas gerações, pela operação denominada elitismo.

A implementação padrão de algoritmo genético para seleção natural é a roleta. Após o processo de seleção, os indivíduos são usados como entrada para outros ope-

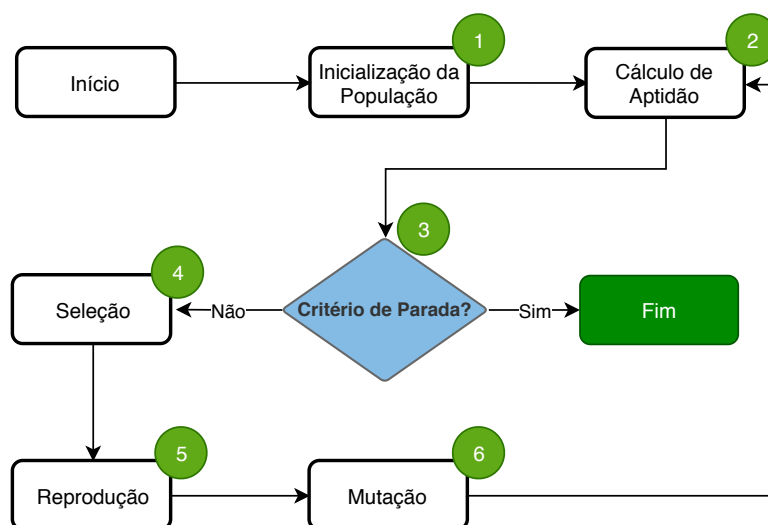
radores genéticos: *crossover* e mutação. O operador *crossover* combina duas cadeias de cromossomos e a mutação modifica alguns bits em um único cromossomo. Normalmente, para um problema particular o algoritmo genético necessita das seguintes quatro componentes (MICHALEWICZ, 1996):

- Uma representação genética para potenciais soluções do problema.
- A inicialização de uma população atual.
- Uma função de aptidão.
- Operadores genéticos para alterar a estrutura dos filhos.

3.2 Fluxograma

O fluxograma da Figura 4 identifica a sequência de passos realizados pelo algoritmo. O primeiro passo (1) de um algoritmo genético é a criação de uma população inicial de forma aleatória, composta por um conjunto de indivíduos que representam as possíveis soluções para o problema. O segundo passo é o cálculo de aptidão (2), onde a população é avaliada e cada indivíduo recebe uma nota (denominada de aptidão) determinando a adaptabilidade do indivíduo. O terceiro passo é verificar se o critério de parada (3) foi atingido. O quarto passo é selecionar os indivíduos (4) utilizando os operadores de seleção para a etapa de reprodução (5), onde os indivíduos combinam seus materiais genéticos gerando novos indivíduos descendentes. O último passo é a mutação (6) onde uma pequena taxa da população¹ sofre alteração nos genes. Após essa etapa, uma nova população é criada, retornando ao passo 2.

Figura 4 – Fluxograma.



Fonte: Próprio Autor.

¹ Normalmente utiliza-se algo entre 0.1% a 5%.

3.3 Algoritmo

Algoritmo 1 Algoritmo Genético

```
1: procedimento GA
2:   Inicialize  $t = 0$ , onde  $t$  representa a geração
3:   Gera a população Inicial  $P(t)$ , randomicamente
4:   para cada indivíduo  $i \in P(t)$  faça
5:     Avalie a função de fitness
6:   fim para
7:   enquanto não alcançar o critério de parada faça
8:     Selecione alguns indivíduos de  $P(t)$ 
9:     Aplique a operação de crossover nos indivíduos selecionados
10:    Aplique mutação nos indivíduos selecionados
11:    Aplique elitismo nos indivíduos selecionados
12:     $t = t + 1$ 
13:    Avalie a função de fitness para a nova população  $P(t)$ 
14:  fim enquanto
15: devolve Melhor Indivíduo ou solução
16: fim procedimento
```

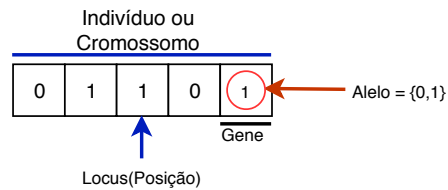
3.4 Terminologia

A teoria da evolução de *Darwin* indica que o meio ambiente seleciona a cada geração os seres vivos mais aptos/resistentes de uma população para sobrevivência. Resultando em que os mais resistentes conseguem reproduzir e gerar descendentes, enquanto os menos aptos tendem a ser eliminados durante as gerações. Dessa forma, é realizado o mapeamento dos termos encontrados na biologia com os utilizados pelos algoritmos genéticos (LACERDA; CARVALHO, 1999):

- **Cromossomo:** Representa a estrutura de dados que codifica a solução para um determinado problema, ou seja, representa um simples espaço de busca.
- **Gene:** Representa a característica, ou seja, é um parâmetro codificado no cromossomo (elemento do vetor).
- **Indivíduo:** Um membro da população, podendo ser formado por um (tradicional) ou mais cromossomos.
- **Genótipo:** Representa a informação contida no cromossomo.
- **Fenótipo:** Representa a estrutura, e é construído a partir das informações do genótipo. É basicamente o cromossomo decodificado.
- **Alelo:** Representa os valores que o gene pode assumir.
- **Locus:** É a posição do gene no cromossomo.
- **Aptidão(*fitness*):** é uma nota atribuída a cada indivíduo, ou seja, determina a adaptabilidade de um indivíduo em um ambiente. Também denominada de função de aptidão.

A Figura 5 indica um exemplo com algumas terminologias.

Figura 5 – Terminologia.



Fonte: Próprio autor.

3.5 Função de Aptidão

Em uma população existe um número finito de indivíduos, de modo que cada indivíduo possui uma aptidão (*fitness*), ou seja, uma nota que irá determinar quão bom é aquele indivíduo. É interessante que a medida de aptidão seja bastante representativa, diferenciando de forma correta os indivíduos mais aptos dos menos aptos (HAMDANI et al., 2011).

3.6 Representação Genética

Ao modelar um problema é interessante verificar qual codificação do indivíduo utilizar, pois, cada problema possui sua especificidade e codificação necessária. Existem problemas como o do caixeiro viajante² que não permitem indivíduos com codificação binária, pois, os valores dos genes em um cromossomo precisam ser distintos. Já o problema da mochila³ necessita da codificação binária (mais tradicional). A Figura 6 demonstra algumas possíveis codificações.

² O problema do caixeiro viajante têm como objetivos encontrar uma menor rota partindo de uma cidade de origem e retornando a ela, passando pelas cidades adjacentes somente uma única vez.

³ No problema da mochila existe um número limitado de itens para colocar em uma mochila levando em consideração o peso de cada item e a capacidade da mochila. A ideia é maximizar o valor total dos itens, sem ultrapassar o limite de peso máximo.

Figura 6 – Representações Genéticas.

Ind 1	0	1	1	0	0	1	0	1	Codificação Binária					
Ind 2	3	2	6	4	1	8	7	5	Codificação Inteira					
Ind 3	2.0	1.5	2.8	4.7	8.7	1.2	2.4	4.9	Codificação Real					
Ind 4	a	r	t	n	v	q	p	i	Codificação por Caracteres					
Ind 5	<table><tr><td>^</td><td>v</td><td>></td><td><</td><td>>></td></tr></table>								^	v	>	<	>>	Codificação por Comandos
^	v	>	<	>>										

Fonte: Próprio Autor.

3.7 Métodos de Seleção

Nesta seção serão apresentados métodos tradicionais utilizados pelos algoritmos genéticos para selecionar os indivíduos para a etapa de reprodução. O processo é realizado de forma determinística com base na função de aptidão de cada indivíduo.

3.7.1 Seleção por Roleta

Seleção por Roleta é um dos métodos mais tradicionais para seleção de indivíduos em uma população. Uma abordagem de seleção comum atribui uma probabilidade de seleção P_j para cada indivíduo j com base em seu valor de aptidão (PENICHEVA; ATANASSOV; SHANNON, 2009). A ideia é selecionar indivíduos baseado em uma probabilidade que é diretamente proporcional ao valor de aptidão, ou seja, a seleção de um indivíduo corresponde a uma fatia da roleta. Os indivíduos que possuem maiores aptidões (maior fatia da roleta) têm maiores probabilidades de serem escolhidos, onde o indivíduo mais apto ocupa o maior segmento enquanto o menos apto possui um menor segmento correspondente dentro da roleta. A circunferência da roleta é a soma de todos os valores de aptidão dos indivíduos (RAZALI; GERAGHTY et al., 2011). A probabilidade P do i -ésimo indivíduo é dada por:

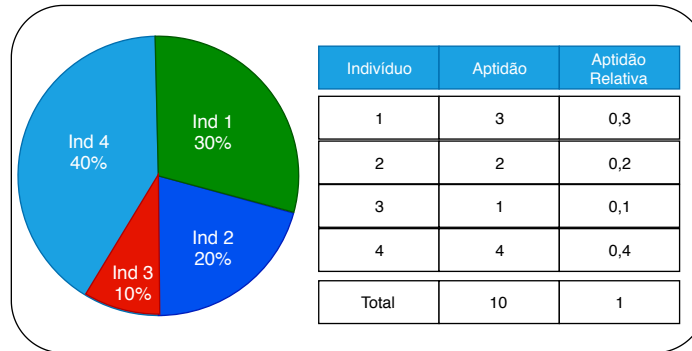
$$P(i) = \frac{fitness(i)}{\sum_{k=1}^n fitness(k)} \quad (3.1)$$

onde, $fitness$ representa a função de aptidão do indivíduo i e n é o tamanho da população. A sequência de passos pode ser visualizada a seguir:

- Seja $T = \sum_{k=1}^n fitness(c_k)$ o valor da soma das aptidões da população.
- Obtenha um valor aleatório $r \in [0, T]$.
- Percorra toda a população e some as aptidões.
- Quando a soma for maior que r , pare e retorne o indivíduo atual.

A Figura 7 demonstra um exemplo com uma população fictícia e seus respectivos valores.

Figura 7 – Seleção por Roleta.

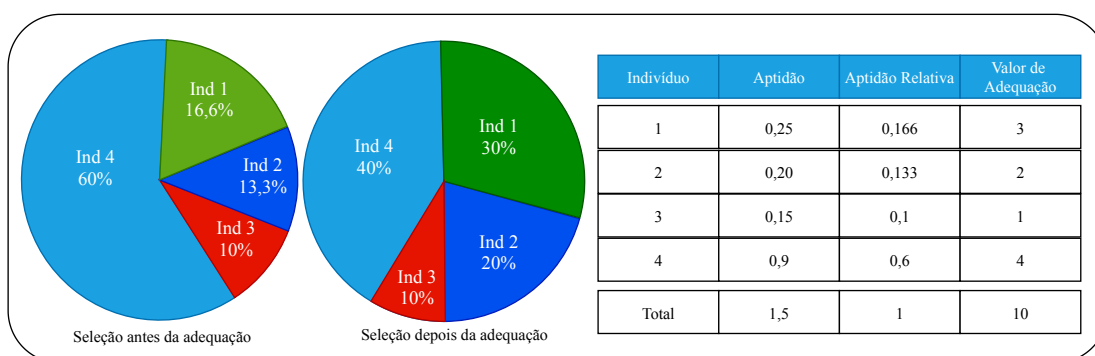


Fonte: Próprio Autor.

3.7.2 Seleção por *Rank*

No método de seleção por roleta existe um problema onde os indivíduos com maiores aptidões têm grandes possibilidades de serem selecionados muitas vezes, desfavorecendo os indivíduos com menores aptidões e consequentemente causando perda da diversidade da população. Na seleção por classificação (*rank*) existe um valor de adequação para cada indivíduo, que é dado em forma crescente com relação ao valor de aptidão relativa, onde o pior indivíduo tem valor de adequação 1 e o melhor tem valor N . Esse processo resulta em convergência lenta, mas ajuda a prevenir a convergência prematura do algoritmo devido a super indivíduos (RAZALI; GERAGHTY et al., 2011), podendo resultar em uma solução de máximo/mínimo local. Após o valor de adequação definido, monta-se novamente a roleta de modo que há agora uma melhor distribuição dos indivíduos, ou seja, indivíduos com baixas aptidões relativas possuem agora mais chances de serem escolhidos.

Figura 8 – Seleção por *Rank*.



Fonte: Próprio Autor.

3.7.3 Seleção por Torneio

Esse método de seleção é baseado no trabalho não publicado por Wetzel e especificado por Brindle (1980) em sua dissertação. A ideia é simples, inicialmente escolhem-se de forma aleatória e de igual probabilidade N indivíduos de uma determinada população P (com ou sem substituição). Após essa etapa, os N^4 indivíduos selecionados realizam uma disputa, onde o indivíduo com melhor aptidão será o escolhido (GOLDBERG; DEB, 1991).

Figura 9 – Seleção por Torneio.



Fonte: Próprio Autor.

3.7.4 Elitismo

Introduzido pela primeira vez por Jong (1975), é uma adição a muitos métodos de seleção que força o AG para manter alguns dos melhores indivíduos em cada geração. Tais indivíduos podem ser perdidos se não forem selecionados para reproduzir ou se eles são destruídos por cruzamento, ou mutação. Muitos pesquisadores descobriram que o elitismo melhora significativamente o desempenho do AG.

3.8 Reprodução

Crossover é uma técnica de reprodução que utiliza dois cromossomos pais e produz dois cromossomos filhos (WRIGHT, 1991).

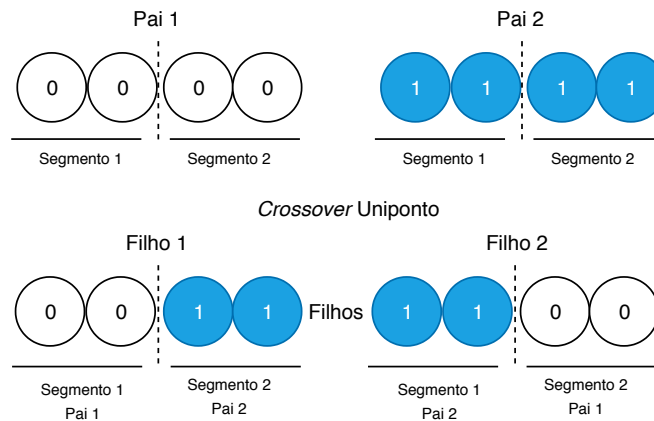
3.8.1 *Crossover* Uniponto

Um comum método utilizado para *crossover* é denominado *crossover* uniponto. Nesse método, ambos os pais são divididos em subcromossomos a esquerda e à direita de cada pai com tamanhos iguais. Então, cada filho obtém o subcromossomo à esquerda de

⁴ Normalmente utilizam-se 3 indivíduos.

um pai, e o subcromossomo à direita de outro pai. A divisão (entre dois sucessivos genes) é denominada ponto de corte (WRIGHT, 1991). A Figura 10 demonstra com maiores detalhes a operação com um indivíduo binário.

Figura 10 – *Crossover* Uniponto.

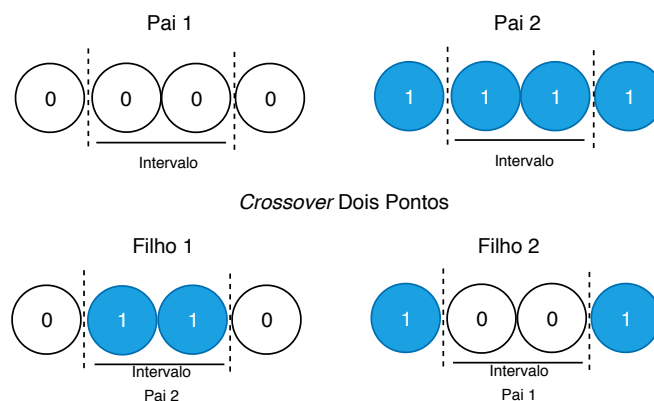


Fonte: Próprio Autor.

3.8.2 *Crossover* dois pontos

Neste modelo, existem dois pontos de corte selecionados aleatoriamente, sendo que o intervalo do ponto de corte será utilizado para a troca de materiais genéticos entre os indivíduos. A Figura 11 ilustra um exemplo.

Figura 11 – *Crossover* Dois Pontos.



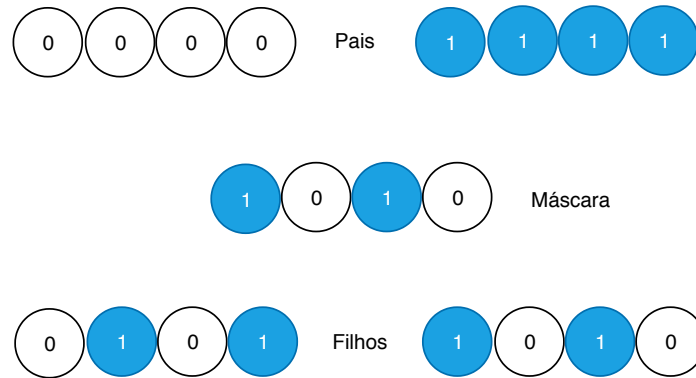
Fonte: Próprio Autor.

3.8.3 *Crossover* Uniforme

No *Crossover* uniforme (SPEARS; JONG, 1991), existe uma máscara binária com quantidade de genes iguais ao tamanho dos indivíduos, onde o valor 0 representa que o

filho 1 recebe o gene do pai 2 e o valor 1 representa que o filho 2 recebe o gene do pai 1. É possível visualizar na Figura 12 a máscara e os filhos gerados.

Figura 12 – *Crossover* Uniforme.



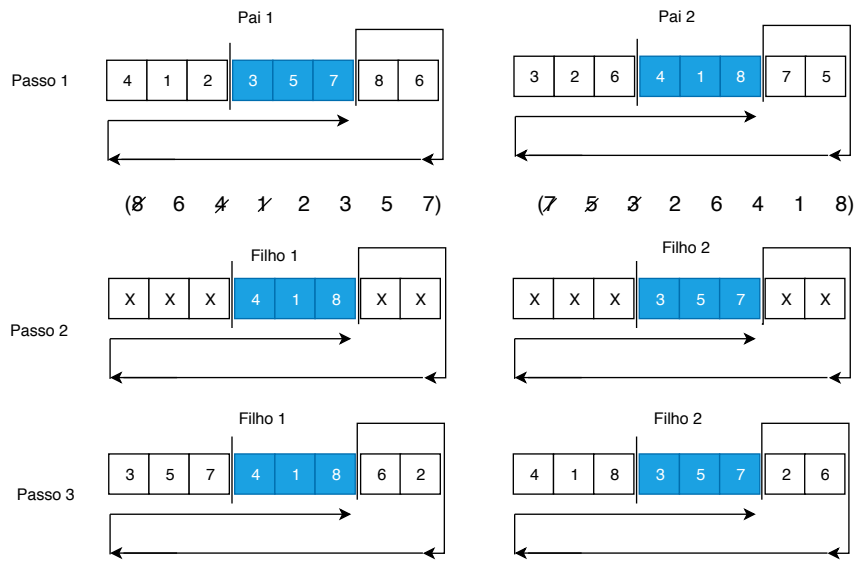
Fonte: Próprio Autor.

3.8.4 *Order Crossover*

Segundo Goldberg e Holland (1988), o *crossover* recebe dois pais denominados de pai 1 e pai 2 e gera dois novos descendentes denominados de filho 1 e filho 2. Aleatoriamente, um subconjunto de genes é mapeado para os respectivos filhos, mantendo as mesmas posições para os novos indivíduos. Esse método pode ser utilizado em problemas que necessitam da distinção dos valores dos genes, ou seja, não pode haver genes com valores iguais. Basicamente o funcionamento segue da seguinte forma:

- Define-se o intervalo de corte de forma aleatória.
- O indivíduo filho 1 deverá receber os genes do pai 2 no mesmo intervalo de corte.
- Os genes do pai 1 com valores iguais aos do filho 1 são removidos.
- Após o ponto de corte do filho 1, os valores serão preenchidos com base na ordem do pai 1 após o ponto de corte sendo análoga na rotação de bits.

A execução é equivalente para o filho 2. A Figura 13 derivada de Alves, Oliveira e Rocha Neto (2017) demonstra a sequência de passos.

Figura 13 – *Order Crossover*.

Fonte: Próprio Autor.

3.9 Muta  o

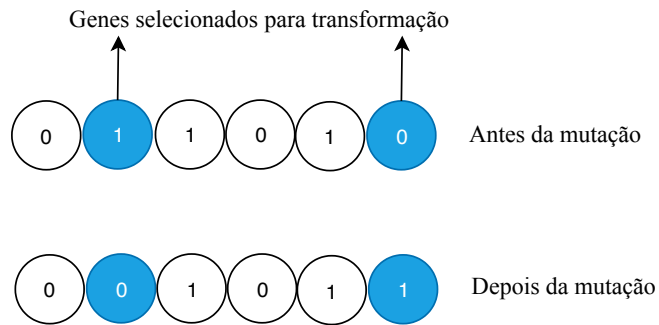
A muta  o   necess ria, pois, mesmo atrav s da reprodu  o e cruzamento, ocasionalmente eles podem se tornar super zelosos e perder algum material gen tico potencialmente  til (GOLDBERG, 1989). Nesta etapa, uma pequena taxa da popula  o   afetada pela muta  o, oferecendo uma componente aleat ria, com o intuito de inserir uma nova caracter stica na popula  o ou reinserir uma caracter stica que pode ter sido perdida nos processos de reprodu  o. Por fim, a muta  o tamb m objetiva fugir de m nimos/m ximos locais. Essa rotina evita a perda da diversidade da popula  o (HOLLAND et al., 1992) e assegura uma explora  o mais ampla do espa o de busca. A taxa de muta  o sobre uma popula  o tem probabilidade previamente estabelecida.

A muta  o tamb m pode destruir alguma informa  o contida no cromossomo. Logo   interessante utilizar uma taxa de muta  o pequena⁵, mas suficiente para garantir a diversidade da popula  o (LACERDA; CARVALHO, 1999). Os principais operadores de muta  o s o demonstrados a seguir.

3.9.1 Bit Flip

Esse operador alterna aleatoriamente alguns dos bits em um cromossomo. Por exemplo, a cadeia 00000100 pode ser mutada em sua segunda posi  o para produzir 01000100. A muta  o pode ocorrer em cada posi  o de bit em uma string com alguma probabilidade, geralmente muito pequena (MITCHELL, 1998). A Figura 14 ilustra um exemplo.

⁵ Normalmente entre 0,1% a 5%.

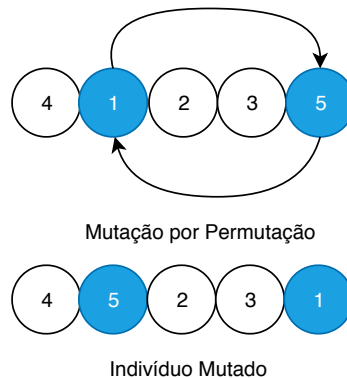
Figura 14 – Muta  o *bit-flip*.

Fonte: Pr  prio Autor.

3.9.2 Muta  o por Permuta  o (*Swap*)

Neste modelo,    escolhido de forma aleat  ria genes para troca do material gen  tico, de modo que eles permutam o valor como mostra a Figura 15.

Figura 15 – Muta  o por permuta  o.



Fonte: Pr  prio Autor.

3.10 Crit  rios de Parada

O algoritmo gen  tico possui alguns crit  rios de parada para encerrar a busca, como por exemplo (MICHALEWICZ, 1996):

- **N  mero de Gera  es:** O algoritmo finaliza sua execu  o quando n  mero m  ximo de gera  es predefinido    atingido.
- **Converg  ncia da Fun  o Objetivo:** A chance de alcan  ar mudan  as significativas nas pr  ximas gera  es    excessivamente baixa, ou seja, quando a diferen  a entre a m  dia das aptid  es da popula  es (anterior e atual) for menor que um valor *epsilon* predefinido, o algoritmo termina.

- **Sem mudança no melhor valor de aptidão:** Quando o valor de aptidão do melhor indivíduo não altera durante um determinado número de gerações a busca é encerrada.
- **Valor ótimo conhecido:** Quando já se conhece o valor ótimo, o algoritmo finaliza ao encontrá-lo.

3.11 Conclusão

Neste capítulo foram apresentados os fundamentos teóricos dos algoritmos genéticos, bem como estrutura, operadores de reprodução e mutação. É interessante salientar que os AGs realizam buscas simultâneas em várias regiões do espaço de busca durante o processo baseado na teoria da evolução de *Darwin*, pois trabalham com uma população de indivíduos. São guiados pela função de aptidão e bem fáceis de serem implementados em computadores.

A codificação a ser utilizada é intrínseca ao problema de otimização a ser solucionado, logo a correta codificação é de suma importância. Devido suas características, é mais simples resolver problemas com AGs ao utilizar métodos matemáticos que necessitam de linearidade, diferenciabilidade, continuidade ou convexidade da função objetivo.

A mutação é bastante importante para escapar de mínimos/máximos locais, inserir uma nova característica ou até mesmo reinserir uma característica que tenha sido perdida ao longo das gerações.

4 Obtenção de Modelos Esparsos e Redução de Dimensionalidade

Como mencionado, o classificador LSSVM sofre com a falta de esparsidade, portanto, neste capítulo serão apresentados métodos para obtenção de classificadores esparsos inspirados em redução de vetores-suporte e seleção de atributos.

4.1 Métodos para Redução de Vetores-Suporte

Vetores-Suporte constituem uma pequena parcela dos dados de treinamento, devendo ser os únicos a serem armazenados durante o processo de treinamento. Como mencionado anteriormente, a LSSVM apresenta uma desvantagem importante em termos de esparsidade com relação ao SVM (SUYKENS; LUKAS; VANDEWALLE, 2000), a partir da escolha da norma 2 e consequentemente pelo fato de que os valores de suporte são proporcionais aos erros nos pontos de dados, em $\alpha_i = \gamma \xi_i$. Diante disto, vários métodos foram desenvolvidos ao longo dos anos com o intuito de impor esparsidade no algoritmo. Carvalho e Braga (2009) citam alguns métodos como *Pruning* LSSVM e LS²-SVM.

4.1.1 *Opposite Maps*

O método *Opposite Maps* (OM) (ROCHA NETO; BARRETO, 2013) tem como objetivo a obtenção de um conjunto reduzido a partir do total de vetores de treinamento. Após o processamento do algoritmo, são obtidos padrões de treinamento que são fortes candidatos à pertencerem ao conjunto de vetores-suporte. A sequência de passos é definida a seguir:

- PASSO 1 - Dividir o conjunto de dados de treinamento em dois subconjuntos, um para cada classe.
- PASSO 2 - Treinar uma rede SOM (*Self-Organizing Maps*) (KOHONEN, 2001) para cada subconjunto. As redes são denominadas SOM-1 e SOM-2.
- PASSO 3 - Encontrar o neurônio vencedor correspondente da rede SOM-1. Por conseguinte, deve-se podar os neurônios mortos da rede SOM-1. Repetir o processo para a rede SOM-2.
- Para cada neurônio ativado anteriormente, encontram-se K pontos mais próximos, encontrando assim o conjunto reduzido de vetores candidatos a pertencerem ao conjunto de vetores-suporte.

4.1.2 LS²-SVM

O método LS²-SVM (VALYON; HORVÁTH, 2004) elimina as colunas de $\mathbf{W}$ sem remover as linhas correspondentes. Basicamente, o algoritmo é dado em duas fases que tenta reduzir $\mathbf{W}$ com o intuito de detectar os vetores-suporte. A primeira fase é realizada reduzindo a dimensão da matriz $\mathbf{W}$ sendo apenas um algoritmo de eliminação de colunas. O objetivo é realizar operações elementares na matriz $\mathbf{W}$ afim de obter uma matriz escalonada reduzida $\mathbf{W}^*$. Um limiar ϵ é aplicado a $\mathbf{W}^*$, onde os elementos que são menores que o valor de ϵ são setados para zero. Após obter a matriz reduzida $\mathbf{W}^*$, as colunas correspondentes que possuem apenas elementos com valores zero são eliminadas. Todas as linhas de $\mathbf{W}^*$ são mantidas, enquanto algumas colunas são removidas, de modo que $\mathbf{W}^*$ não é quadrada. O segundo passo então é resolver o sistema linear $\mathbf{W}^*\mathbf{z} = \mathbf{l}$, utilizando a pseudo-inversa para que se consiga obter a solução $\mathbf{z}$. Este método possui um parâmetro extra ϵ que corresponde a tolerância utilizada pelo processo de redução.

4.1.3 *Pruning* LSSVM

O método P-LSSVM proposto por Suykens, Lukas e Vandewalle (2000) avalia os valores absolutos dos multiplicadores de Lagrange (ordenados) $|\alpha_i|$ para identificar quais dados são mais significantes para a contribuição da LSSVM. A eliminação dos multiplicadores de Lagrange ocorre de forma recursiva, onde a cada iteração são removidos alguns vetores de treinamento (normalmente 5%) do conjunto de vetores-suporte. Suykens, Lukas e Vandewalle (2000) propuseram uma remoção gradual do tamanho do conjunto de vetores-suporte. Onde o critério de parada é dado pela diminuição do desempenho do classificador no conjunto de validação. A remoção dos vetores-suporte é feita pela associação do valor do multiplicador de Lagrange a 0, sem alterações das dimensões da matriz. Seguem os devidos passos realizados pelo algoritmo para a remoção de padrões menos relevantes no conjunto de treinamento.

- PASSO 1: Treinar um classificador LSSVM com todo o conjunto de Treinamento
- PASSO 2: Remover uma pequena quantidade dos vetores-suporte (normalmente 5%) pertencentes ao conjunto de treinamento, em que os valores de $|\alpha_i|$ ordenados sejam os menores.
- PASSO 3: Retreinar o classificador LSSVM usando o método da inversa Eq. (2.13) com o conjunto de treinamento reduzido.
- PASSO 4: Voltar ao PASSO 2, ao considerar que a performance no conjunto de validação não diminua. Caso contrário, finalizar o processo com o conjunto de treinamento da iteração anterior.

4.1.4 IP-LSSVM

O método IP-LSSVM (poda) foi proposto por Carvalho e Braga (2009) e incorpora as vantagens dos classificadores SVM e LSSVM, como detecção automática de vetores-suporte e uma solução obtida simplesmente pela solução de sistemas de equações lineares. Este método avalia diretamente o valor do multiplicador de Lagrange α_i e não seu valor absoluto (CARVALHO; BRAGA, 2009). A abordagem indica que os padrões que estão próximos da superfície de separação e longe dos hiperplanos são suscetíveis a serem os vetores-suporte, o que corresponde a $\alpha_i \gg 0$. Os novos critérios de relevância propostos por Carvalho e Braga (2009) podem ser descritos por:

- $\mathbf{x}_i$ com $\alpha_i \gg 0$ é classificado como vetor-suporte. De modo que $\mathbf{x}_i$ está localizado na região de fronteira entre as duas classes ou na região da classe oposta. O que corresponde aos vetores não-nulos associados α_i quando comparados com o SVM solucionado por problemas de programação quadrática.
- $\mathbf{x}_i$ com $\alpha_i \geq 0$ é removido. Então α_i é classificado corretamente perto do hiperplano de apoio, correspondendo aos vetores associados com $\alpha_i = 0$ no SVM.
- $\mathbf{x}_i$ com $\alpha_i < 0$ ou $\alpha_i \ll 0$ será eliminado. Sendo que $\mathbf{x}_i$ é corretamente classificado, longe do hiperplano de decisão, o que corresponde aos vetores associados com $\alpha_i = 0$ nos classificadores SVM.

O critério proposto é aplicado ao IP-LSSVM para eliminar colunas não relevantes da matriz original $\mathbf{W}$ e para construir uma matriz não quadrada $\mathbf{W}_2$ que será utilizada na segunda etapa. As colunas removidas correspondem aos vetores-suporte menos relevantes para o problema de classificação, selecionados de acordo com seu valor do multiplicador de Lagrange. As linhas de $\mathbf{W}$ não são removidas, pois sua eliminação levaria a uma perda de informação de rótulos e também em performance (VALYON; HORVÁTH, 2004).

O primeiro passo é realizado utilizando a função inversa para resolver o sistema de equações lineares representadas pela Eq. (2.9). A solução desse sistema é dada pela Eq. (2.11), sendo $\mathbf{W}$ uma matriz de tamanho $N_p + 1$, onde N_p é o número de pontos de treinamento.

No segundo passo, um sistema $\mathbf{W}_2 \mathbf{z}_2 = \mathbf{l}$ é resolvido utilizando o método da pseudo-inversa para resolução do sistema linear após a remoção das colunas de $\mathbf{W}$ pela Eq. (2.15), cuja solução $\mathbf{z}_2$ corresponde ao *bias* (primeiro elemento) e os multiplicadores de Lagrange α_i associados aos vetores de treinamento $\mathbf{x}_i$. Vale salientar que o valor de b (*bias*) não será removido. O processo de treinamento proposto por Carvalho e Braga (2009) pode ser descrito por:

- Treinar um classificador LSSVM com todo o conjunto de treinamento por $\mathbf{z} = \mathbf{W}^{-1} \mathbf{l}$ pois $\mathbf{W}$ é uma matriz quadrada.
- Especificar um parâmetro $\tau \in [0, 1]$, que define o percentual dos vetores de

treinamento que serão vetores-suporte.

- Ordenar os vetores de treinamento por seus valores de α_i .
- O percentual $(1-\tau)$ dos vetores de treinamento que correspondem aos menores valores de α_i são selecionados.
- Construir uma matriz não quadrada $\mathbf{W}_2$ pela remoção das colunas de $\mathbf{W}$ associadas aos vetores com menores valores de α_i .
- Resolver o novo sistema linear $\mathbf{z}_2 = \mathbf{W}^\dagger \mathbf{1}$, onde $\mathbf{W}^\dagger = (\mathbf{W}_2^T \mathbf{W}_2)^{-1} \mathbf{W}_2$.
- Os vetores-suporte são os vetores do treinamento associados às colunas presentes na matriz $\mathbf{W}_2$.
- Por fim, os valores dos α 's e b são obtidos de $\mathbf{z}_2$.

O classificador esparso IP-LSSVM possui os mesmos parâmetros de treinamento da LSSVM, o parâmetro de regularização γ e os parâmetros da função de *kernel*. Existe também um outro parâmetro, τ , que corresponde a fração de vetores de treinamento que serão selecionados como vetores-suporte. Contanto que o valor de τ não seja crítico, é bem fácil ajustar após informar o valor de γ .

4.2 Classificadores Esparsos Baseados em Algoritmos Genéticos

Algoritmos genéticos (AGs) são vastas técnicas de busca/otimização. A estrutura de potenciais soluções é codificada por uma representação cromossômica. Uma população de cromossomos é evoluída através de conceitos derivados da genética e seleção natural (operadores de seleção), com o objetivo de produzir a cada geração indivíduos mais resistentes (GOLDBERG, 1989; HAUPT; HAUPT, 2004).

Normalmente, os algoritmos genéticos são utilizados para encontrar os parâmetros ótimos da função de *kernel*, por exemplo, ou também ajustar os parâmetros específicos do classificador (MUSTAFA et al., 2012; SAMADZADEGAN; SOLEYMANI; ABBASPOUR, 2010).

Diferentemente de trabalhos anteriores que utilizam AGs para encontrar parâmetros ótimos do classificador e/ou *kernel*, os métodos propostos por esta dissertação combinam AGs e LSSVM para encontrar soluções esparsas. De forma similar, existem trabalhos presentes na literatura que removem vetores-suporte para alcançar esparsidade (SILVA; ROCHA NETO, 2013; SILVA; SILVA; ROCHA NETO, 2015; SILVA; ROCHA NETO, 2015).

4.2.1 GAS-LSSVM

O trabalho denominado GAS-LSSVM (*Genetic Algorithm for Sparse LSSVM*) proposto por Silva e Rocha Neto (2013) tem como base a utilização de algoritmos genéticos

para a eliminação de vetores-suporte da LSSVM, removendo padrões do treinamento que são irrelevantes, ou seja, existe pouca contribuição para a classificação sem afetar a acurácia do classificador (SILVA; ROCHA NETO, 2013). O método utiliza um indivíduo binário com tamanho N_p , que é a quantidade de vetores-suporte presentes no conjunto de treinamento. Se o gene do cromossomo possuir valor '1' o vetor-suporte irá permanecer no conjunto de dados e se for '0' será eliminado. O algoritmo tem como objetivo maximizar a acurácia e reduzir a quantidade de vetores-suporte. Essa abordagem consegue alcançar esparsidade pelo fator de remover determinado padrão do conjunto de treinamento.

De modo semelhante com o que acontece com o algoritmo IP-LSSVM, o método proposto por Silva e Rocha Neto (2013) remove apenas as colunas da matriz $\mathbf{W}$, que é definido pelos multiplicadores de Lagrange associados setados para valor '0' de acordo com a codificação do melhor indivíduo. Se a k -ésima linha é removida, então uma restrição do problema é eliminada, o que pode acarretar perda de informação e desempenho (VALYON; HORVÁTH, 2004).

4.2.2 MOGAS-LSSVM

Denominado *Multi-Objective Genetic Algorithm for Sparse LSSVM* (MOGAS-LSSVM), este método propõe uma nova abordagem de redução de vetores-suporte, onde cada indivíduo é codificado de forma semelhante à abordagem anterior (Seção 4.2.1) definindo a manutenção de um padrão com respectivo gene com valor '1' e remoção do padrão com gene associado ao valor '0' (SILVA; ROCHA NETO, 2014). A função de *fitness* visa maximizar a acurácia e minimizar o número de vetores-suporte, mas o processo de otimização leva em consideração um custo Θ para a poda de padrões, como pode ser visualizado na Eq (4.1).

$$fitness = 1 - \left\{ \frac{(1 - \text{acurácia}) + ((1 - \Theta) * \text{redução})}{2 - \Theta} \right\} \quad (4.1)$$

onde $\Theta \in [0, 1]$ é um parâmetro exclusivo do classificador, acurácia é a taxa de acerto sobre o conjunto de treinamento (validação), já a redução é dada por:

$$\text{redução} = \frac{\#\text{número de padrões podados}}{\#\text{padrões do treinamento}} \quad (4.2)$$

Quando o custo Θ possui valor baixo (0), então os valores de *fitness* dependem da acurácia e passam a ter uma proporção direta com a redução. Já com o valor de $\Theta = 1$ o classificador se torna de objetivo simples, guiado apenas pela acurácia, abordagem equivalente ao método GAS-LSSVM anteriormente demonstrado na Seção 4.2.1.

4.2.3 FSGAS-LSSVM

O classificador *Fixed-Size Genetic Algorithm for Sparse LSSVM* (FSGAS-LSSVM) tem como base um algoritmo genético de objetivo simples e com genes codificados como binário. Para que se consiga um tamanho fixo de vetores-suporte que serão podados, algumas modificações nas operações de reprodução e mutação foram realizadas (SILVA; ROCHA NETO, 2015).

Nessa proposta, o classificador FSGAS-LSSVM seleciona um tamanho fixo de genes que serão os vetores-suporte utilizados na fase de classificação. O objetivo é remover padrões não relevantes para o problema ou que podem conter ruídos dificultando o desempenho do classificador.

A população inicial é gerada com um tamanho fixo de genes com valor '1' devendo respeitar o parâmetro de poda necessário para o algoritmo, estando no intervalo $\in [0, 1]$ que representa o percentual de vetores-suporte que serão eliminados.

4.2.3.1 Reprodução

Como mencionado anteriormente, o operador de reprodução precisa manter a quantidade de '1s' nos genes iguais após as operações. Para que isso aconteça, uma codificação indireta foi utilizada de acordo com os passos a seguir.

- 1 - Escolher de forma aleatória um subconjunto de genes.
- 2 - É realizada uma codificação indireta de cada pai para uma nova representação através de um indivíduo com codificação inteira.
- 3 - Os indivíduos codificados são submetidos ao operador OX (Ver Seção 3.8.1)
- 4 - Conversão dos indivíduos obtidos após o processo de reprodução é realizada para a representação binária correspondente.

4.2.3.2 Mutação

O indivíduo do classificador FSGAS-LSSVM possui genes com valores '0' ou '1', de modo que o operador de mutação utiliza um par de genes para troca de material genético.

4.3 Métodos para Seleção de Atributos

Seleção ou extração de atributos constituem uma parte importante do contexto de aprendizado de máquina, reconhecimento de padrões e mineração de dados, sendo um tema de pesquisa e desenvolvimento em evolução desde a década de 1970 (LEE, 2005) e representa um papel primordial na fase de pré-processamento dos dados (KOHAVI; JOHN, 1997; FAYYAD; PIATETSKY-SHAPIRO; SMYTH, 1996). Com a seleção de atributos, o custo computacional tende a diminuir e o desempenho da classificação pode aumentar. Uma

representação adequada dos dados de todas as funcionalidades é um problema importante em aprendizado de máquina. Todas as características originais do problema podem nem sempre ser importantes para as tarefas de classificação e regressão. Algumas características podem ser irrelevantes/redundantes ou conter ruídos, o que pode atrapalhar na tarefa de classificação, de modo que para aumentar o desempenho do classificador e reduzir o custo computacional deve-se utilizar o processo de seleção de atributos (CAO et al., 2007).

Com a seleção de atributos, é possível por exemplo, construir um *ranking* ordenado de atributos seguindo algum critério de importância, a redução de dimensionalidade do espaço de busca e a redução de características que contém ruídos. No problema de seleção de subconjuntos de características, um algoritmo de aprendizado enfrenta o problema de selecionar alguns subconjuntos de atributos nos quais concentra sua atenção, ignorando o resto (KOHAVI; JOHN, 1997). A seleção de atributos pode ser importante, por exemplo, em situações onde a medição de atributos é custosa, permitindo que um subconjunto representativo e menor que o original seja escolhido (LEE, 2005).

Vale salientar que com os dados melhorados, consequentemente é possível que o desempenho dos algoritmos de aprendizado sejam aperfeiçoados de forma significativa (LEE, 2005). Algumas vantagens da seleção de características são apresentadas a seguir (SILVA, 2011):

- Expectativa de geração de modelos de dados mais compreensíveis, ao selecionar um subconjunto reduzido;
- Economia de espaço em memória e de operações computacionais tornam o processo mais eficiente;

Diante disto, são propostos métodos para seleção de atributos do conjunto de dados com o intuito de impor esparsidade ao classificador LSSVM, com métodos com objetivo simples e tamanho fixo que serão exemplificados no Capítulo 5.

4.3.1 Análise de Componentes Principais

A Análise de componentes principais (*Principal Component Analysis*, PCA) (JOLIFFE, 2002) é uma famosa técnica de redução de dimensionalidade não-supervisionada bastante utilizada na biologia, reconhecimento de imagens, processamento de sinais, reconhecimento facial e compressão de imagens (BARBER, 1980; THOMAZ; GIRALDI, 2010; ZHAO et al., 1998; DU; FOWLER, 2007). Da perspectiva estatística de reconhecimento de padrões, o valor prático de componentes principais é que fornece uma efetiva técnica para redução de dimensionalidade (HAYKIN et al., 2009).

Com o PCA, é possível identificar padrões nos dados e expressá-los de maneira a destacar suas semelhanças e diferenças. Em dados com alta dimensão onde a representação gráfica não está disponível, pode ser difícil encontrar esses padrões. Portanto, o PCA é

uma ferramenta útil para analisar os dados. Uma das vantagens do PCA é que, depois de encontrar padrões nos dados, você compacta os dados, por exemplo, reduzindo o número de dimensões, sem muita perda de informação. A sequência de passos pode ser vista abaixo (SMITH, 2002).

- Obtenha os dados.
- Normalize os dados.
- Calcule a matriz de covariância com base na Eq. (4.3).
- Calcule os autovetores e autovalores da matriz de covariância.
- Escolha os componentes principais e forme um vetor de atributos.
- Obtenha o novo conjunto de dados.

4.3.1.1 Redução de Dimensionalidade

Considere um conjunto de dados $\mathbf{Z} \in \mathbb{R}^{p \times n}$, onde p é o número de amostras e n a quantidade de atributos. De modo que o PCA tem como base a variabilidade presente nos dados, o primeiro passo então é calcular a matriz de covariância. Ela é utilizada para avaliar quanto os atributos variam da média em relação aos outros (OLIVEIRA, 2016). É possível calcular a covariância de duas variáveis aleatórias $\mathbf{x}$ e $\mathbf{y}$ com base na Eq. (4.3).

$$\text{cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{p-1} \sum_{i=1}^p (x_i - \bar{x})(y_i - \bar{y}) \quad (4.3)$$

onde $\bar{x}$ é a média, dada pela Eq. (4.4).

$$\bar{x} = \frac{1}{p} \sum_{i=1}^p x_i \quad (4.4)$$

É possível calcular as entradas da matriz de covariância de $\mathbf{Z}$, dada por uma matriz quadrada $\mathbf{R}_{i,j} = \text{cov}(\mathbf{x}_i, \mathbf{x}_j)$ onde $\mathbf{R} \in \mathbb{R}^{n \times n}$. Segundo Haykin et al. (2009) a melhor motivação para utilização do PCA é a redução de dimensionalidade, cuja essência é resumida nas Eqs (4.5) e (4.6). Seja $\lambda_1, \lambda_2, \dots, \lambda_l$ os l maiores autovalores de $\mathbf{R}$. É possível realizar a representação dos dados tendo como base um vetor $\mathbf{x}$ n -dimensional, então $\mathbf{x}$ pode ser representado por um l -dimensional vetor de componentes principais, denotado pela Eq. (4.5).

$$a_j = \mathbf{q}_j^T \mathbf{x} = \mathbf{x}^T \mathbf{q}_j, \quad j = 1, 2, \dots, l$$

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_l \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_l^T \end{bmatrix} \mathbf{x}, \quad l \leq n \quad (4.5)$$

onde, $\mathbf{q}_i$ é o i -ésimo autovetor e a_i , o i -ésimo componente do vetor $\mathbf{a}$, é a projeção do vetor de dados $\mathbf{x}$ para o i -ésimo autovetor $\mathbf{q}_i$. Vale salientar que os autovetores de $\mathbf{R}$ são escolhidos com base no valor do seu respectivo autovalor. De modo que quanto maior for o autovalor, maior é a variância presente no autovetor correspondente. Portanto, é possível encontrar as l componentes principais apenas ordenando os autovalores (λ) e os respectivos autovetores ($\mathbf{q}$) e escolhendo os l autovetores com base nos l maiores valores de λ . Se $l = n$, então o novo vetor $\mathbf{a}$ é a versão rotacionada do vetor de dados originais $\mathbf{x}$, a diferença real entre eles é que $\mathbf{a}$ possui componentes não correlacionadas. Caso $l < n$, então apenas um subconjunto dos autovetores é utilizado o que torna a representação de dados aproximada, podendo ser chamado de redução de dimensionalidade.

Também é possível realizar a reconstrução dos dados, por exemplo, dado um vetor de componentes principais $\mathbf{a}$ então, o vetor de dados original $\mathbf{x}$ pode ser reconstruído por uma combinação linear dos autovetores como mostra a Eq. (4.6).

$$\hat{\mathbf{x}} = \sum_{j=1}^l a_j \mathbf{q}_j, \quad l \leq n \quad (4.6)$$

de modo que, os componentes principais $a_1, a_2, \dots, a_l$ são os coeficientes da expansão. A reconstrução será exata se $l = n$, e aproximada se $l < n$. O vetor de erro resultante é dado por:

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}} \quad (4.7)$$

Uma aplicação bastante utilizada com PCA é a compressão de dados. Nesta aplicação, o objetivo é preservar muita informação sobre um conjunto de entrada de dados possível. Dado um vetor de dados $\mathbf{x}$ de dimensão n , PCA alcança o objetivo através do subespaço de decomposição dos dados de entrada em que os primeiros l (menores que n) componentes principais dos dados de entrada fornecem um mapeamento linear (HAYKIN et al., 2009).

O mapeamento é ótimo quando permite a reconstrução dos dados de entrada originais no sentido de erro mínimo quadrático médio. Além disso, uma representação baseada nas primeiras componentes principais é mais aconselhável a uma representação arbitrária, pois as componentes principais dos dados de entrada são naturalmente ordenadas, diminuindo o autovalor ou, equivalentemente, diminuindo a variância. Assim, é possível otimizar o uso da análise de componentes principais para compressão de dados, empregando a maior precisão numérica para codificar o primeiro componente principal da entrada e progressivamente menos precisão para codificar os demais componentes $l - 1$ (HAYKIN et al., 2009).

Um exemplo a ser ilustrado na base de dados *Vertebral Column Pathologies*, utiliza apenas uma amostra do conjunto de dados para mostrar o funcionamento do PCA. Na Tabela 2 é exibida uma pequena amostra do conjunto de dados original, na Tabela 3 os

dados foram então normalizados, na Tabela 4 a matriz de covariância obtida é apresentada, já na Tabela 5 são apresentados os autovetores referentes aos atributos: ângulo de incidência pélvica (IP), versão pélvica (VP), ângulo de lordose (AL), declive sacral (DS), raio pélvico (RP) e grau de deslizamento (ROCHA NETO, 2011). Os autovalores são demonstrados na Tabela 6 e por fim, o percentual de informação retido pelas componentes principais pode ser visualizado na Tabela 7.

Tabela 2 – Amostra dos dados da coluna vertebral.

IP	VP	AL	DS	RP	GD
46,37	10,22	42,70	36,16	121,25	-0,54
44,43	14,17	32,24	30,26	131,72	-3,60
52,42	19,01	35,87	33,41	116,56	1,69
51,33	13,63	33,26	37,69	131,31	1,79
63,77	12,76	65,36	51,01	89,82	56,00

Fonte: Próprio Autor.

Tabela 3 – Amostra dos dados normalizados da coluna vertebral.

IP	VP	AL	DS	RP	GD
-0,819	-0,731	-0,497	-0,506	0,250	-0,714
-0,932	-0,337	-1,061	-0,945	1,036	-0,796
-0,468	0,146	-0,865	-0,711	-0,102	-0,655
-0,531	-0,391	-1,006	-0,392	1,005	-0,652
0,189	-0,477	0,723	0,600	-2,110	0,790

Fonte: Próprio Autor.

Tabela 4 – Matriz de covariância.

1,000	0,629	0,717	0,815	-0,247	0,638
0,629	1,000	0,432	0,062	0,032	0,397
0,717	0,432	1,000	0,598	-0,080	0,533
0,815	0,062	0,598	1,000	-0,342	0,523
-0,247	0,032	-0,080	-0,342	1,000	-0,026
0,638	0,397	0,533	0,523	-0,026	1,000

Fonte: Próprio Autor.

Tabela 5 – Autovetores da Coluna Vertebral.

0,717	-0,423	0,102	-0,096	-0,002	0,535
-0,416	-0,150	0,006	-0,648	0,527	0,323
0,000	0,676	0,548	0,152	0,092	0,458
-0,558	-0,431	0,127	0,360	-0,396	0,445
-0,000	-0,275	0,174	0,586	0,727	-0,143
0,000	0,278	-0,801	0,271	0,162	0,424

Fonte: Próprio Autor.

Tabela 6 – Autovalores da Coluna Vertebral.

0,000	0,326	0,473	0,761	1,195	3,246
-------	-------	-------	-------	-------	-------

Fonte: Próprio Autor.

Tabela 7 – Percentual de Informação em cada componente principal.

PC1	PC2	PC3	PC4	PC5	PC6
54,10%	19,91%	12,68%	7,88%	5,43%	0,00%

Fonte: Próprio Autor.

4.3.1.2 PCA-LSSVM

O Método proposto por Çalişir e Dogantekin (2011) introduz um classificador LSSVM para o sistema de diagnóstico de hepatite. O sistema manipula a combinação de extração e classificação de características. Após o processo, a quantidade de 19 atributos foi reduzida para apenas 10, mantendo a taxa de acurácia do classificador original.

Os autores justificam o uso do PCA no trabalho proposto, ao abordar que os atributos devem ser reduzidos (isto é, do conjunto de dados original) para uma menor dimensão por um extrator de características. O vetor de características reduzido obtido inclui informações mais úteis do que o vetor de características original. O trabalho é dividido em duas partes como pode ser visto abaixo:

- Extração e redução de características do conjunto de dados por PCA.
- Classificação com o LSSVM.

4.3.2 Análise de Discriminantes Lineares

A análise de discriminantes lineares (*Linear Discriminant Analysis*, LDA) (FUKU-AGA, 1990) é um método bastante conhecido no quesito de extração de atributos e redução

de dimensionalidade, comumente utilizado em diversas áreas tais como reconhecimento facial (BELHUMEUR; HESPANHA; KRIEGMAN, 1997) e classificação de dados de micro-arranjos (DUDOIT; FRIDLYAND; SPEED, 2002). Basicamente, a LDA projeta os dados em um espaço vetorial de dimensão mais baixa, de modo que a razão entre a distância entre classes e a distância dentro da classe seja maximizada, atingindo discriminação máxima (YE; JANARDAN; LI, 2005). Para a opção de redução de dimensionalidade, o parâmetro número de componentes deverá ser menor ou igual a $(num_classes - 1)$, onde $num_classes$ é o número de classes do problema.

4.4 Conclusão

Como abordado anteriormente, ao longo dos anos, vários métodos foram propostos com o intuito de resolver o problema de falta de esparsidade do classificador LSSVM. Portanto, neste capítulo foram abordados alguns classificadores presentes na literatura para redução de vetores-suporte com modelos reduzidos, bem como redução de dimensionalidade. Em alguns métodos, algoritmos genéticos foram utilizados para obtenção de soluções esparsas. Esses métodos visam obter uma superfície de decisão o mais semelhante possível da original.

Métodos como P-LSSVM, IP-LSSVM e PCA-LSSVM foram abordados, bem como utilizados como propostas de comparação. As vantagens da seleção de atributos em um conjunto de dados original, garante ganhos em eficiência no quesito de espaço em memória e custo computacional. Também é dissertado sobre a análise de componentes principais, bem como sua fundamentação teórica e funcionamento para redução de dimensionalidade.

5 Métodos Propostos

Neste capítulo são apresentados três métodos desenvolvidos para obtenção de modelos reduzidos e consequentemente impondo esparsidade ao classificador LSSVM sendo eles: *FPGAS-LSSVM*(Proposta 1), *FSFGAS-LSSVM*(Proposta 2) e *PFGAS-LSSVM*(Proposta 3) utilizando algoritmos genéticos para encontrar o indivíduo que possui melhor acurácia.

5.1 FPGAS-LSSVM (Proposta 1)

A proposta denominada *Feature Pruning Genetic Algorithm for Sparse LSSVM*, FPGAS-LSSVM é baseada em um algoritmo genético simples que tem como objetivo maximizar a acurácia do classificador removendo atributos do conjunto de dados. Basicamente, o indivíduo é representado por um vetor binário, em que '1' representa que o atributo permanecerá no treinamento (validação e teste) do conjunto de dados enquanto '0' representa a ausência desse atributo (ARAÚJO; ROCHA NETO, 2019). A função de aptidão é baseada no maior valor de acurácia obtido pelo classificador no conjunto de validação levando em conta os genes configurados com '1'.

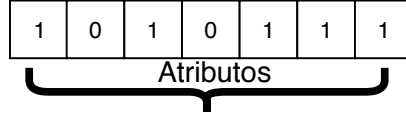
Nessa abordagem, a acurácia é obtida pelo classificador após a remoção dos atributos, a fim de remover características que não são relevantes para o problema e ainda mantém ou mesmo melhora a precisão da classificação. Vale salientar que geralmente os conjuntos de dados possuem mais padrões que atributos, refletindo em maior impacto quando um atributo é eliminado, pois evita que sejam salvos N_p valores, onde N_p é o tamanho do conjunto de treinamento.

5.1.1 Indivíduos ou Cromossomos

Nesta proposta, o indivíduo é descrito por um vetor $\mathbf{g} = [g_1 \ g_2 \ \dots \ g_i \ \dots \ g_{N_f-1} \ g_{N_f}]$, em que $g_i \in \{0,1\}$ e N_f é o número de atributos do conjunto de dados. Sempre que $g_i = 0$, significa que o i -ésimo atributo estará fora do conjunto de treinamento (validação e teste). Também é possível dizer que o número de características podadas será a quantidade de zeros presentes no vetor $\mathbf{g}$.

A Figura 16 ilustra um exemplo de indivíduo com a remoção do segundo e quarto atributos (colunas). Para demonstrar a abordagem proposta, é apresentado de forma simplificada um exemplo de como modelar um indivíduo para o FPGAS-LSSVM. Considerando a matriz $\mathbf{B}$ como um conjunto de dados de exemplo composto por 7 atributos, isto significa que um indivíduo representava um vetor de genes [1111111].

Figura 16 – Indivíduo.



Fonte: Próprio Autor.

$$\mathbf{B} = \begin{bmatrix} 0,4378 & 0,1326 & 0,2037 & 0,8998 & 0,4694 & 0,8705 & 0,4578 \\ 0,2802 & 0,5450 & 0,5444 & 0,2179 & 0,4138 & 0,6030 & 0,7222 \\ 0,9852 & 0,8278 & 0,8749 & 0,0770 & 0,5027 & 0,2653 & 0,3390 \\ 0,6088 & 0,8370 & 0,1210 & 0,4742 & 0,1254 & 0,8648 & 0,4012 \\ 0,2537 & 0,8333 & 0,8564 & 0,8350 & 0,1323 & 0,0581 & 0,5270 \end{bmatrix} \quad (5.1)$$

A Matriz $\mathbf{B}_1$ representa o resultado da poda de atributos do conjunto de dados original com base no indivíduo presente na Figura 16. Em geral, para eliminar determinado atributo do conjunto de dados original que é descrito pelo i -ésimo gene (g_i) no vetor $\mathbf{g} = [g_1 \ g_2 \ g_3 \ g_4 \ g_5 \ g_6 \ g_7]$, é necessário remover i -ésima coluna.

$$\mathbf{B}_1 = \begin{bmatrix} 0,4378 & 0,2037 & 0,4694 & 0,8705 & 0,4578 \\ 0,2802 & 0,5444 & 0,4138 & 0,6030 & 0,7222 \\ 0,9852 & 0,8749 & 0,5027 & 0,2653 & 0,3390 \\ 0,6088 & 0,1210 & 0,1254 & 0,8648 & 0,4012 \\ 0,2537 & 0,8564 & 0,1323 & 0,0581 & 0,5270 \end{bmatrix} \quad (5.2)$$

É interessante reforçar a ideia de que a poda de atributos não interfere na matriz $\mathbf{W}$, de modo que ela continua em sua forma quadrada, sendo necessário apenas utilizar o método da inversa presente na Eq. (2.13) para obter o *bias* e os multiplicadores de Lagrange.

A redução é calculada de forma simplificada de acordo com a quantidade de atributos removidos como segue na Eq. (5.3):

$$redução = \frac{\#Quantidade\ de\ Atributos\ Removidos}{\#Quantidade\ de\ Atributos} \quad (5.3)$$

5.1.2 Algoritmo

Basicamente, a ideia do algoritmo é realizar a poda dos atributos do conjunto de dados de acordo com a codificação presente cada indivíduo. A aptidão da população é avaliada com o conjunto de validação $\mathbf{Z}_2$ e por fim, o melhor indivíduo será utilizado como solução, obtendo a taxa de acerto no conjunto de teste.

Algoritmo 2 FPGAS**Entradas:** Total de gerações: t_{max} , Tamanho da população: Q e Conjunto de Dados: $\mathbf{Z}$.**Saída:** Melhor indivíduo da população P_{new} .

```

1:  $t \leftarrow 0$  ▷ Inicia o contador de gerações
2:  $P(t) \leftarrow geraPopulacaoInicial(Q)$ 
3: para  $i=1$  até  $\leftarrow size(P(t))$  faça
4:   Remover os atributos de  $\mathbf{Z}$  de acordo com o gene = 0 do indivíduo  $i$  de  $P(t)$ 
5:   Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$  ▷ Treino, Validação e Teste
6:   Treinar LSSVM com  $\mathbf{Z}_1$ 
7:   Avaliar a função de fitness no conjunto de validação  $\mathbf{Z}_2$  por (5.4)
8: fim para
9: enquanto  $t < t_{max}$  faça
10:    $t = t + 1$ ;
11:    $P_{new} \leftarrow P.selecaoTorneio()$ 
12:    $P_{new} \leftarrow P.reproducao()$ 
13:    $P_{new} \leftarrow P.mutacao()$ 
14:    $P_{new} \leftarrow P.elitismo()$ 
15:   para  $i=1$  até  $\leftarrow size(P_{new})$  faça
16:     Remover os atributos de  $\mathbf{Z}$  de acordo com o gene = 0 do indivíduo  $i$  de  $P(t)$ 
17:     Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$  ▷ Treino, Validação e Teste
18:     Treinar LSSVM com  $\mathbf{Z}_1$ 
19:     Avaliar a função de fitness em  $\mathbf{Z}_2$  para a nova população  $P_{new}$  por (5.4);
20:      $P \leftarrow P_{new}$ 
21:   fim para
22: fim enquanto
  Avaliar o fitness do melhor indivíduo da população  $P_{new}$  no conjunto de teste  $\mathbf{Z}_3$ 
  devolve  $P_{new.melhorIndividuo}()$ 

```

5.1.3 Função de *Fitness*

A ideia principal da função de *fitness* é maximizar a acurácia de um indivíduo candidato a uma solução $\mathbf{s}$, enquanto poda os atributos do conjunto de dados original, então foi formulada a função de *fitness* como uma função objetivo dada por:

$$fitness(\mathbf{s}) = acurácia(\mathbf{s}) \quad (5.4)$$

onde $acurácia(\mathbf{s})$ é calculada como:

$$acurácia(\mathbf{s}) = 1 - erro(\mathbf{s}) \quad (5.5)$$

5.2 FSFGAS-LSSVM (Proposta 2)

Nessa abordagem denominada *Fixed Size Feature Genetic Algorithm for Sparse LSSVM* - FSFGAS-LSSVM a ideia é manter uma taxa de poda fixa de características, diferentemente da abordagem anterior mencionada (Seção 5.1) em que a quantidade de

atributos removidos é determinada pelo melhor indivíduo. Para conseguir a poda de tamanho fixo dos atributos, algumas modificações nos operadores tradicionais do algoritmo genético foram necessárias, em especial os operadores de mutação e reprodução.

É interessante mencionar que diante de todas as gerações, o número de atributos que serão removidos será sempre o mesmo, sendo essa característica o grande diferencial da abordagem aqui mencionada. A função de *fitness* é dada pelo valor resultante da acurácia do classificador quando é levado em conta os genes que possuem valor '1'. Cada indivíduo possui a quantidade de genes igual à quantidade de atributos do conjunto de dados, de modo que se o valor for '0' o respectivo atributo será eliminado.

5.2.1 Indivíduos ou Cromossomos

Esta proposta tem como base um indivíduo que é descrito por um vetor de genes $\mathbf{g} = [g_1 \ g_2 \ \dots \ g_i \ \dots \ g_{N_f-1} \ g_{N_f}]$, onde $g_i \in \{0,1\}$ e N_f é o número de atributos contidos no conjunto de dados. Contudo, quando $g_i = 0$, isto significa que a i -ésima característica estará fora do conjunto de treinamento (validação e teste). De modo que, a quantidade de atributos presentes no novo conjunto de dados (T_f) será o número de 1's existentes no indivíduo como mostra a Eq. (5.6). É possível então dizer que o número de atributos removidos será a quantidade de 0's presentes no vetor $\mathbf{g}$.

$$T_f = \sum_{i=1}^{N_f} g_i. \quad (5.6)$$

Conforme declarado, cada característica eliminada evita salvar N_p valores, onde N_p é o número de padrões no treinamento. De forma simplificada, é possível ilustrar um exemplo do funcionamento do indivíduo para a proposta FSFGAS-LSSVM. Considerando a matriz $\mathbf{T}$ como um exemplo de *dataset* com 5 atributos, isto significa que um indivíduo possuía um vetor de genes igual a [11111].

Neste modelo, o seguinte vetor de genes: [01011] representa a remoção da primeira e terceira colunas (atributos), a matriz $\mathbf{T}_1$ representa o resultado da poda de atributos do conjunto de dados original. Por se tratar de uma proposta com tamanho fixo, existe um custo de poda $\theta \in [0, 1]$ que será o percentual de atributos removidos. Suponha um exemplo com uma base de dados com 40 atributos e $\theta = 0,8$ (80%), a sequência de passos pode ser visualizada a seguir:

- Definir o parâmetro θ que representa o percentual de remoção dos atributos.
- Criar um vetor v_1 de 0's com tamanho $0,8 * 40 = 32$.
- Criar um vetor v_2 de 1's com tamanho $0,2 * 40 = 8$.
- Concatenar os vetores $[v_1 \ v_2]$ em v_{new} .
- Embaralhar os elementos de v_{new} .

$$\mathbf{T} = \begin{bmatrix} 0,8765 & 0,6694 & 0,8218 & 0,7009 & 0,9441 \\ 0,2035 & 0,3182 & 0,3274 & 0,0614 & 0,8074 \\ 0,7155 & 0,1388 & 0,0336 & 0,8988 & 0,9258 \\ 0,3145 & 0,9801 & 0,3240 & 0,3885 & 0,7995 \\ 0,3587 & 0,4151 & 0,6392 & 0,6347 & 0,4518 \end{bmatrix} \quad (5.7)$$

Vale salientar que a população inicial não pode ser inicializada de forma aleatória, devendo ser estabelecida de acordo com o θ previamente escolhido para que se mantenha a mesma quantidade de 0's e 1's ao longo das gerações.

$$\mathbf{T}_1 = \begin{bmatrix} 0,6694 & 0,7009 & 0,9441 \\ 0,3182 & 0,0614 & 0,8074 \\ 0,1388 & 0,8988 & 0,9258 \\ 0,9801 & 0,3885 & 0,7995 \\ 0,4151 & 0,6347 & 0,4518 \end{bmatrix} \quad (5.8)$$

5.2.2 Função de *Fitness* para FSFGAS-LSSVM

A ideia central da função de *fitness* é semelhante a abordagem anterior (Seção 5.1.3) onde visa maximizar a acurácia obtida por um indivíduo ou solução candidata $\mathbf{s}$, enquanto poda os atributos do conjunto de dados original com base em θ .

5.2.3 Operador de Reprodução

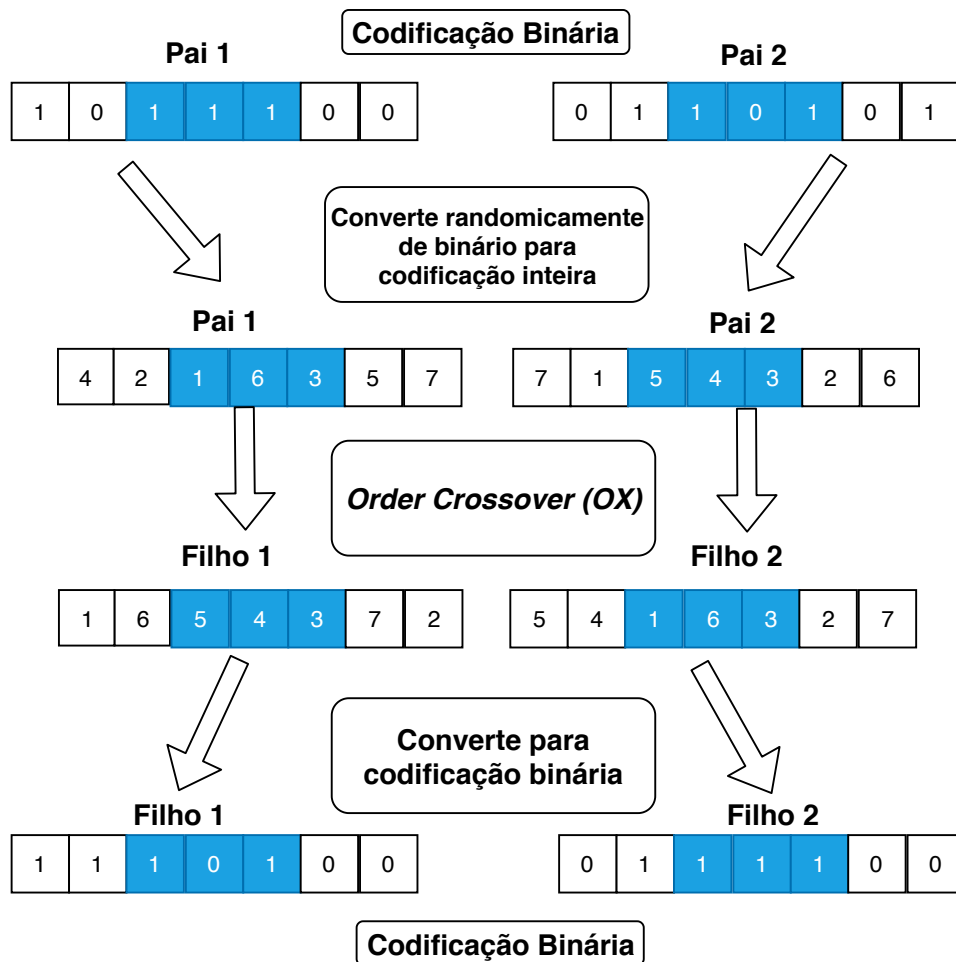
A ideia por trás do operador de reprodução da proposta é manter a mesma quantidade de 1's e 0's no indivíduo após aplicar os operadores genéticos. Essa prática é necessária pois os operadores de reprodução tradicionais podem produzir indivíduos inválidos, ou seja, não havendo números constantes de genes com valor '1' nos indivíduos. Neste caminho, é possível garantir a quantidade fixa de atributos independente das operações realizadas em todos os indivíduos da população, sem a geração de um indivíduo inválido por exemplo.

Portanto, é proposta a utilização de codificação indireta de indivíduos (SILVA; ROCHA NETO, 2015). Inicialmente os indivíduos foram mapeados da codificação binária para uma codificação inteira, o que permite realizar operações de permutação, como *cycle* (CX), *partially matched* (PMX) e *Order Crossover* (OX) (EIBEN; SMITH, 2003). Na Figura 17, derivada de Silva e Rocha Neto (2015), o passo a passo das operações de reprodução são demonstrados, com o mapeamento do indivíduo binário para uma codificação inteira e então aplicando o operador de reprodução OX (Seção 3.8.4). Finalmente, o mapeamento

inverso é realizado para obter um indivíduo binário, de modo que cada filho seja adequado para a próxima população. Um passo a passo pode ser visualizado a seguir:

- Transforme a codificação binária em inteira randomicamente.
- Ponto de corte escolhido aleatoriamente.
- Aplique o operador de reprodução OX.
- Realize o mapeamento entre o indivíduo com codificação inteira para binária.
- Retorne o filho reproduzido.

Figura 17 – Processo de transformação entre codificação binário-inteiro para aplicação do operador OX seguido pela transformação entre a codificação inteiro-binário.



Fonte: Próprio Autor.

5.2.4 Mutação

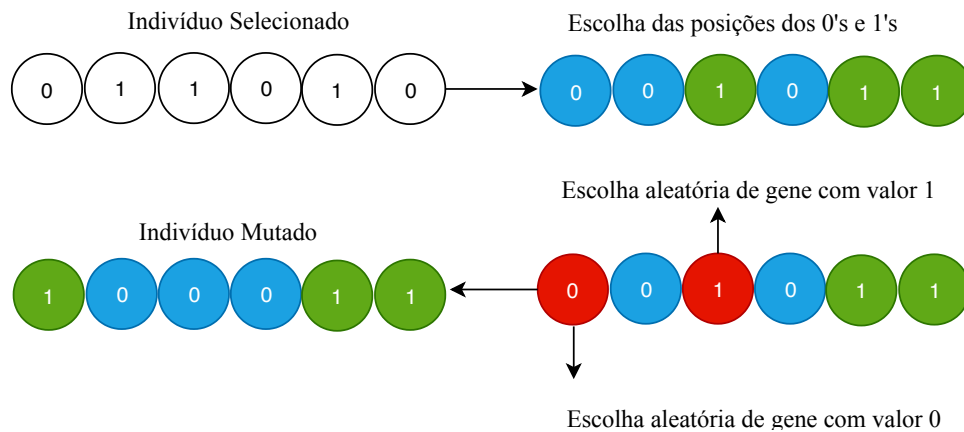
O algoritmo FSFGAS-LSSVM como mencionado anteriormente, possui a característica de poda com tamanho fixo, de modo que a mutação sofreu uma alteração em relação ao modelo tradicional com o intuito de ajustar o funcionamento do algoritmo

para alcançar o objetivo acima mencionado. A sequência de passos presentes na Figura 18 demonstra de maneira intuitiva o funcionamento geral.

Portanto, o operador de mutação *bit flip* (Ver Seção 3.9.1) foi utilizado com o intuito de escapar de mínimos locais, inserir uma nova característica ao indivíduo ou até mesmo reinserir alguma característica que possa ter sido perdida ao longo das gerações. A ideia é dividir o indivíduo em dois subconjuntos, o primeiro com os genes que possuem valor '1' e o segundo com genes que possuem valor '0'. Então são escolhidos aleatoriamente bits de cada subconjunto e invertidos seus respectivos valores. Essa abordagem garante que a quantidade de 0's e 1's seja mantida após a operação de mutação, como segue:

- Recebe o indivíduo.
- Vetor $\mathbf{x}$ com as posições dos 0's presentes no indivíduo.
- Vetor $\mathbf{y}$ com as posições dos 1's presentes no indivíduo.
- Escolhe aleatoriamente uma posição i do vetor $\mathbf{x}$.
- Inverte o valor da posição i do indivíduo para 1.
- Escolhe aleatoriamente uma posição j do vetor $\mathbf{y}$.
- Inverte o valor da posição j do indivíduo para 0.
- Retorna o indivíduo mutado.

Figura 18 – Sequência de passos realizados pela mutação.



Fonte: Próprio Autor.

5.2.5 Algoritmo

O algoritmo dessa proposta é bastante semelhante com o algoritmo 2, com a adição de um parâmetro θ que determina o percentual de poda. Os operadores de reprodução e mutação sofreram ajustes com o intuito de produzirem indivíduos válidos. A população inicial é gerada de acordo com o percentual de poda escolhido.

Algoritmo 3 FSFGAS

Entradas: Total de gerações: t_{max} , Tamanho da população: Q , Taxa de poda: θ e Conjunto de Dados: $\mathbf{Z}$.

Saída: Melhor indivíduo da população: P_{new} .

```

1:  $t \leftarrow 0$                                 ▷ Inicia o contador de gerações
2:  $P(t) \leftarrow \text{geraPopulacaoInicialFixa}(Q, \theta)$                 ▷ Nova Versão
3: para  $i=1$  até  $\leftarrow \text{size}(P(t))$  faça
4:   Remover os atributos de  $\mathbf{Z}$  de acordo com o gene = 0 do indivíduo  $i$  de  $P(t)$ 
5:   Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$                                 ▷ Treino, Validação e Teste
6:   Treinar LSSVM com  $\mathbf{Z}_1$  após a poda
7:   Avaliar a função de fitness em  $\mathbf{Z}_2$  por (5.4)
8: fim para
9: enquanto  $t < t_{max}$  faça
10:    $t = t + 1$ 
11:    $P_{new} \leftarrow P.\text{selecaoTorneio}()$ 
12:    $P_{new} \leftarrow P.\text{reproducaoFixa}()$                                 ▷ Nova Versão
13:    $P_{new} \leftarrow P.\text{mutacaoFixa}()$                                 ▷ Nova Versão
14:    $P_{new} \leftarrow P.\text{elitismo}()$ 
15:   para  $i=1$  até  $\leftarrow \text{size}(P_{new})$  faça
16:     Remover os atributos de  $\mathbf{Z}$  de acordo com o gene = 0 do indivíduo  $i$  de  $P(t)$ 
17:     Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$                                 ▷ Treino, Validação e Teste
18:     Avaliar o fitness no conjunto  $\mathbf{Z}_2$  para cada indivíduo  $i$  por (5.4)
19:      $P \leftarrow P_{new}$ 
20:   fim para
21: fim enquanto
22: Avaliar o fitness do melhor indivíduo de  $P_{new}$  no conjunto  $\mathbf{Z}_3$ 
23: devolve  $P_{new}.\text{melhorIndividuo}()$ 

```

5.3 PFGAS-LSSVM (Proposta 3)

A terceira proposta denominada PFGAS-LSSVM (*Patterns and Features Genetic Algorithm for Sparse LSSVM*) é baseada em um algoritmo genético que visa maximizar a acurácia do classificador unindo as vantagens da seleção de atributos com a detecção de vetores-suporte. Assim, o funcionamento é dado pela remoção de atributos e padrões do conjunto de treinamento. A Tabela 8 ilustra a comparação entre alguns métodos mencionados nesta dissertação.

Nesta abordagem, o indivíduo possui dois cromossomos binários (Figura 19) denominados $\mathbf{c}_1$ e $\mathbf{c}_2$ representando os atributos (Figura 19a) e padrões de treinamento (Figura 19b), respectivamente. No vetor binário $\mathbf{c}_1$, o valor ('1') representa a presença de uma determinada característica no conjunto de dados de treinamento (validação e teste), consequentemente, zero ('0') representa a ausência desse atributo. Para o cromossomo $\mathbf{c}_2$, a ideia é basicamente a mesma, com a diferença de que agora a referência é sobre padrões de treinamento. A função de aptidão será guiada pela acurácia do classificador, levando

Tabela 8 – Comparação de poda entre os classificadores.

Classificador	Poda Atributos	Poda Padrões
P-LSSVM		X
IP-LSSVM		X
GAS-LSSVM		X
FSGAS-LSSVM		X
MOGAS-LSSVM		X
PCA-LSSVM	X	
FPGAS-LSSVM	X	
FSFGAS-LSSVM	X	
PFGAS-LSSVM	X	X

Fonte: Próprio Autor.

em consideração os genes definidos como '1' nos cromossomos $\mathbf{c}_1$ e $\mathbf{c}_2$.

É interessante mencionar que nessa abordagem, cada indivíduo possui tantos atributos e padrões de treinamento quanto o número de genes presentes em $\mathbf{c}_1$ e $\mathbf{c}_2$ configurados como '1'. Se o gene tiver um valor '0', o atributo/padrão será excluído. Existem métodos que constroem a matriz $\mathbf{W}$ da LSSVM para posteriormente remover as colunas¹. O método proposto já constrói a matriz com base nas colunas que serão removidas, ou seja, uma matriz reduzida ($\mathbf{W}^*$). Em seguida, é utilizado o método da pseudo-inversa (2.15) para encontrar a solução. Após o processo de poda, a proposta é baseada na acurácia obtida pelo classificador, a fim de remover atributos/padrões que não são relevantes para o problema e ainda mantém ou até melhora a precisão da classificação.

5.3.1 Indivíduos ou Cromossomos

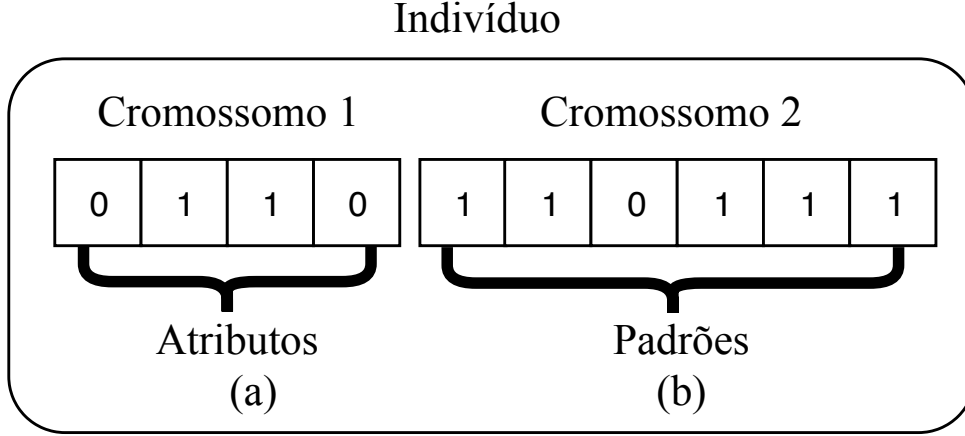
Como declarado, esta proposta possui um indivíduo com dois cromossomos. O primeiro cromossomo é descrito por um vetor $\mathbf{c}_1 = [c_{1_1} \ c_{1_2} \ \dots \ c_{1_{N_f-1}} \ c_{1_{N_f}}]$ representando os atributos e o segundo cromossomo é descrito por um vetor $\mathbf{c}_2 = [c_{2_1} \ c_{2_2} \ \dots \ c_{2_{N_p-1}} \ c_{2_{N_p}}]$ representando os padrões onde $c_{1_i}/c_{2_i} \in \{0,1\}$, N_f é o número de atributos e N_p é o número de padrões de treinamento. O indivíduo i é composto por $\mathbf{c}_1$ e $\mathbf{c}_2$ e tem tamanho $(N_f + N_p)$ como mostrado na Figura 19.

De acordo com a codificação, os genes de $\mathbf{c}_1$ com valor '1', representam os atributos que serão mantidos no novo conjunto de dados. Em contraste, os genes com valor '0' representam os atributos podados. O número de atributos removidos (N_{f_r}) da solução candidata é dada por:

$$N_{f_r} = N_f - \sum_{i=1}^{N_f} c_{1_i}. \quad (5.9)$$

¹ Equivale a setar o multiplicador de lagrange associado ao respectivo padrão do treinamento para 0

Figura 19 – Indivíduo com dois cromossomos representando atributos (a) e padrões (b).



Fonte: Próprio Autor.

Similarmente, o número de vetores-suporte eliminados de acordo com o cromossomo $\mathbf{c}_2$ pode ser calculado por:

$$N_{pr} = N_p - \sum_{j=1}^{N_p} c_{2j}. \quad (5.10)$$

onde N_{pr} representa o número de padrões de treinamento removidos. Alguns trabalhos presentes na literatura utilizam a poda de padrões de treinamento (SILVA; SILVA; ROCHA NETO, 2015; SILVA; ROCHA NETO, 2015; CARVALHO; BRAGA, 2009), já outros trabalhos focam na seleção de atributos (ÇALIŞIR; DOGANTEKIN, 2011; ARAÚJO; ROCHA NETO, 2019), nessa terceira proposta é realizada a combinação entre a seleção de atributos e vetores-suporte. Para realizar o cálculo de redução de elementos do conjunto de treinamento, foi definida uma função denominada (*score*) em percentual utilizada para fins de comparação com o intuito de calcular a redução total obtida por cada classificador como ilustra a Eq. (5.11).

$$score = \frac{(N_p * N_{f_r}) + N_{pr} * (N_f - N_{f_r})}{N_p * N_f} \quad (5.11)$$

Quando $c_{1_i} = 0$, significa que os i -ésimos atributos são eliminados, já para $c_{2_i} = 0$ significa que os i -ésimos padrões estão sendo eliminados do conjunto de treinamento o que equivale a remover a coluna $(i + 1)$ da matriz $\mathbf{W}$. Vale ressaltar que não se deve remover a primeira coluna evitando assim, a perda de desempenho (SILVA; ROCHA NETO, 2013). Então, o número de padrões mantidos é equivalente ao total de 1's presentes em $\mathbf{c}_2$, e o total de atributos é dado de forma similar por $\mathbf{c}_1$. É possível destacar que a quantidade de atributos e padrões removidos é definido como o número de 0's contidos nos vetores $\mathbf{c}_1$ e $\mathbf{c}_2$. Note que o classificador LSSVM necessita salvar todos padrões de treinamento como vetores-suporte, portanto, cada recurso removido evita salvar N_p valores.

Para demonstrar a abordagem proposta, é apresentada de forma simplificada um exemplo de como modelar um indivíduo para PFGAS-LSSVM. Considerando a matriz $\mathbf{M}$ como um conjunto de dados de exemplo com 5 atributos (Eq. 5.12), isto significa que um indivíduo representava um vetor de genes no cromossomo $\mathbf{c}_1 = [11111]$. No modelo proposto, o seguinte vetor de genes $\mathbf{c}_1 = [11110]$ representa a eliminação do quinto atributo (coluna).

$$\mathbf{M} = \begin{bmatrix} 0,7577 & 0,7060 & 0,8235 & 0,4387 & 0,4898 \\ 0,7431 & 0,0318 & 0,6948 & 0,3816 & 0,4456 \\ 0,3922 & 0,2769 & 0,3171 & 0,7655 & 0,6463 \\ 0,6555 & 0,0462 & 0,9502 & 0,7952 & 0,7094 \\ 0,1712 & 0,0971 & 0,0344 & 0,1869 & 0,7547 \end{bmatrix} \quad (5.12)$$

A matriz $\mathbf{M}_1$ representa o resultado da poda de atributos do conjunto de dados original. Em geral, com o intuito de eliminar determinado atributo do conjunto de dados que é descrito pelo i -ésimo gene (c_{1_i}) no vetor $\mathbf{c}_1 = [c_{1_1} \ c_{1_2} \ c_{1_3} \ c_{1_4} \ c_{1_5}]$, é necessário remover a i -ésima coluna.

$$\mathbf{M}_1 = \begin{bmatrix} 0,7577 & 0,7060 & 0,8235 & 0,4387 \\ 0,7431 & 0,0318 & 0,6948 & 0,3816 \\ 0,3922 & 0,2769 & 0,3171 & 0,7655 \\ 0,6555 & 0,0462 & 0,9502 & 0,7952 \\ 0,1712 & 0,0971 & 0,0344 & 0,1869 \end{bmatrix} \quad (5.13)$$

O cromossomo $\mathbf{c}_2$ é responsável pela configuração de poda de vetores-suporte. Se não houvesse a remoção dos padrões do conjunto de treinamento, seria então obtida a matriz $\mathbf{W}$ da LSSVM (Eq. 5.14) com base no conjunto reduzido (Eq. 5.13). Contudo o método proposto realiza a poda dos padrões para que se obtenha então a matriz $\mathbf{W}^*$ já em seu tamanho reduzido.

$$\mathbf{W} = \begin{bmatrix} 0,0000 & 1,0000 & 1,0000 & -1,0000 & -1,0000 & 1,0000 \\ 1,0000 & 21,9432 & 1,3251 & -1,0896 & -1,6606 & 0,3086 \\ 1,0000 & 1,3251 & 21,1816 & -0,8127 & -1,4522 & 0,2255 \\ -1,0000 & -1,0896 & -0,8127 & 20,9170 & 1,1799 & -0,2480 \\ -1,0000 & -1,6606 & -1,4522 & 1,1799 & 21,9670 & -0,2980 \\ 1,0000 & 0,3086 & 0,2255 & -0,2480 & -0,2980 & 20,0749 \end{bmatrix} \quad (5.14)$$

Para remover o primeiro e terceiro padrões, o vetor $\mathbf{c}_2$ deve possuir os seguintes genes: $[01011]$. Há uma diferença com relação a configuração de poda do cromossomo $\mathbf{c}_2$: agora

para eliminar determinado padrão do conjunto de vetores-suporte é necessário remover a $(i + 1)$ coluna.

$$\mathbf{W}^* = \begin{bmatrix} 0,0000 & 1,0000 & -1,0000 & 1,0000 \\ 1,0000 & 1,3251 & -1,6606 & 0,3086 \\ 1,0000 & 21,1816 & -1,4522 & 0,2255 \\ -1,0000 & -0,8127 & 1,1799 & -0,2480 \\ -1,0000 & -1,4522 & 21,9670 & -0,2980 \\ 1,0000 & 0,2255 & -0,2980 & 20,0749 \end{bmatrix} \quad (5.15)$$

Como a matriz $\mathbf{W}^*$ não é quadrada, não é possível obter diretamente a inversa, então para obter a solução é necessário utilizar o método da pseudo-inversa (2.15).

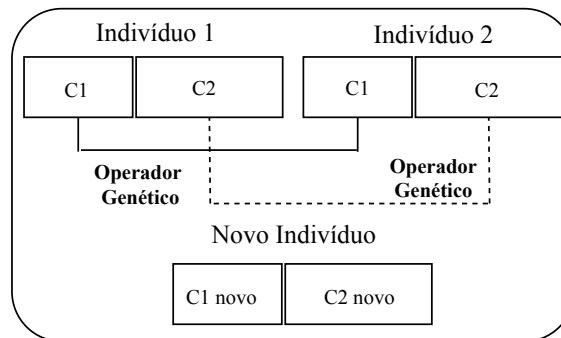
5.3.2 Função de *Fitness* para PFGAS-LSSVM

Como nos métodos mencionados nas seções 5.1 e 5.2, a função de aptidão maximiza a acurácia do classificador com objetivo simples, levando em consideração a poda de atributos e padrões pela Eq. (5.4).

5.3.3 Reprodução

Na reprodução, foi utilizado o *crossover* uniponto. Para esta abordagem, como o indivíduo possui dois cromossomos ($\mathbf{c}_1$ e $\mathbf{c}_2$), a proposta executa a operação mencionada acima separadamente para cada cromossomo, ou seja, dois indivíduos aplicarão os operadores genéticos nos cromossomos equivalentes que produzem um novo indivíduo para a população como mostrado na Figura 20.

Figura 20 – Dois indivíduos realizando operações genéticas nos cromossomos equivalentes.



Fonte: Próprio Autor.

5.3.4 Mutação

O operador de mutação *bit flip* foi utilizado com o intuito de escapar de mínimos locais e inserir alguma nova característica ao indivíduo ou eventualmente reinserir alguma

característica que tenha sido perdida ao longo das gerações.

5.3.5 Algoritmo

O algoritmo 4 representa os procedimentos realizados pela proposta.

Algoritmo 4 PFGAS

Entradas: Total de gerações: t_{max} , Tamanho da população: Q e Conjunto de Dados: $\mathbf{Z}$.

Saída: Melhor indivíduo da população P_{new}

```

1:  $t \leftarrow 0$  ▷ Inicia o contador de gerações
2:  $P(t) \leftarrow geraPopulacaoInicial(Q)$ 
3: para  $i=1$  até  $\leftarrow size(P(t))$  faça
4:   Remover os atributos de  $\mathbf{Z}$  de acordo com  $\mathbf{c}_1$  ▷ Poda de atributos
5:   Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$  ▷ Treino, Validação e Teste
6:   Construir a matriz reduzida  $\mathbf{W}_i^*$  de acordo com  $\mathbf{c}_2$ 
7:   Treinar LSSVM de acordo com  $\mathbf{c}_1$  e  $\mathbf{c}_2$  por (2.15) com  $\mathbf{Z}_1$ 
8:   Avaliar a função de fitness no conjunto de validação  $\mathbf{Z}_2$  por (5.4)
9: fim para
10: enquanto  $t < t_{max}$  faça
11:    $t = t + 1$ ;
12:    $P_{new} \leftarrow P.selecaoTorneio()$ 
13:    $P_{new} \leftarrow P.reproducao(\mathbf{c}_1, \mathbf{c}_2)$ 
14:    $P_{new} \leftarrow P.mutacao(\mathbf{c}_1, \mathbf{c}_2)$ 
15:    $P_{new} \leftarrow P.elitismo()$ 
16:   para  $i=1$  até  $\leftarrow size(P_{new})$  faça
17:     Remover os atributos de  $\mathbf{Z}$  de acordo com  $\mathbf{c}_1$  ▷ Poda de atributos
18:     Dividir  $\mathbf{Z}$  em  $\mathbf{Z}_1$ ,  $\mathbf{Z}_2$  e  $\mathbf{Z}_3$ 
19:     Construir a matriz reduzida  $\mathbf{W}_i^*$  de acordo com  $\mathbf{c}_2$ 
20:     Treinar LSSVM de acordo com  $\mathbf{c}_1$  e  $\mathbf{c}_2$  por (2.15) com  $\mathbf{Z}_1$ 
21:     Avaliar a função de fitness no conjunto de validação por (5.4)
22:      $P \leftarrow P_{new}$ 
23:   fim para
24: fim enquanto
25: Avaliar o fitness do melhor indivíduo da população  $P_{new}$  no conjunto de teste  $\mathbf{Z}_3$ 
   devolve  $P_{new.melhorIndividuo()}$ 

```

5.4 Conclusão

Neste capítulo foram apresentados os métodos propostos por esta dissertação, mais especificamente: FPGAS-LSSVM (Proposta 1), FSFGAS-LSSVM (Proposta 2) e PFGAS-LSSVM (Proposta 3), todos com o intuito em comum de impor esparsidade ao classificador LSSVM. Diante disto, a ideia é obter um conjunto reduzido, de modo a simplificar a estrutura do modelo.

Como já mencionado anteriormente, as duas propostas iniciais têm como base a eliminação de atributos do conjunto de dados, pois obtém maior impacto que a remoção de

padrões. Já a terceira proposta é caracterizada por unir a poda de padrões já presente na literatura em conjunto com a poda de atributos, conseguindo obter valores consideráveis de redução sem prejudicar a acurácia do classificador. O método é baseado em algoritmos genéticos que possuem indivíduos com dois cromossomos aplicados às Máquinas de Vetores-Suporte por Mínimos Quadrados. Vale salientar que algumas modificações em procedimentos padrões do algoritmo genético foram necessárias com o intuito de permitir por exemplo poda com tamanho fixo (FSFGAS-LSSVM) mesmo após as permutações ao longo das gerações.

6 Simulações Computacionais

Toda a metodologia utilizada nas etapas de treinamento, validação e teste será mencionada neste capítulo. As bases de dados utilizadas e resultados obtidos pelos classificadores propostos por esta dissertação serão demonstrados. Para concluir, uma análise sobre os resultados obtidos será realizada bem como uma comparação entre os métodos do estado da arte e os métodos propostos.

6.1 Metodologia

As bases de dados utilizadas para fins de testes e experimentos deste trabalho foram obtidas do repositório *UCI Machine Learning*¹, considerando problemas do mundo real com as respectivas bases: *Diabets*, *Sonar Mines vs. Rocks*, *Bupa Liver Disorders*, *Hepatitis*, *Australian Credit Approval* e *Vertebral Column Pathologies*.

Como o classificador LSSVM foi utilizado para problemas de apenas duas classes, o *dataset Vertebral Column Pathologies* (VCP) foi modificado para um problema binário transformando as classes *hérnia de disco* e *espondilolistese* em apenas uma classe, pois originalmente o *dataset* possui 3 classes. O restante dos *datasets* possuem por padrão apenas duas classes. A Tabela 9 contém as informação de cada base de dados, tais como número de padrões e atributos.

O primeiro passo a ser executado para qualquer conjunto de dados é efetuar a normalização fazendo com que valores medidos em escalas diferentes sejam ajustados para a mesma escala. Para a realização deste procedimento, existem diversos tipos de normalização existentes tais como: *min-max*, *z-score*² e normalização por escalonamento decimal (HAN; PEI; KAMBER, 2011). Nos experimentos realizados foi utilizada a normalização dos dados

¹ <https://archive.ics.uci.edu/ml/>

² Conhecida como normalização de média zero

Tabela 9 – *Datasets* utilizados.

<i>Dataset</i>	Abreviação	Padrões	Atributos
<i>Pima Indians Diabets</i>	PID	768	8
<i>Bupa Liver Disorders</i>	BLD	345	6
<i>Sonar, Mines vs. Rocks</i>	SON	208	60
<i>Hepatitis</i>	HEP	142	19
<i>Australian Credit Approval</i>	AUS	690	14
<i>Vertebral Column Pathologies</i>	VCP	310	6

Fonte: Próprio Autor.

em função da média zero em que os valores de $\mathbf{x}_i$ são normalizados com base em sua média e desvio padrão, como demonstrado a seguir:

$$\mathbf{x}_{i_{norm}} = \frac{\mathbf{x}_i - \mu_x}{\sigma_x} \quad (6.1)$$

Sendo que $\mathbf{x}_i$ representa o i -ésimo padrão do conjunto de dados, a média de todo o conjunto de dados é representada por μ_x , o desvio padrão de todo o conjunto de dados é caracterizado por σ_x e por fim, $\mathbf{x}_{i_{norm}}$ será o padrão normalizado.

Para as simulações propostas, duas configurações distintas foram realizadas de acordo com o tipo de classificador utilizado. Na configuração 1, os dados de treinamento, validação e teste levam em consideração um conjunto de dados completo $\mathbf{Z}$, que foi dividido aleatoriamente em $\mathbf{Z}_1$, $\mathbf{Z}_2$ e $\mathbf{Z}_3$ com 60%, 20% e 20% das amostras para treinamento, validação e teste, respectivamente. Desta forma, o conjunto de dados $\mathbf{Z} = \mathbf{Z}_1 \cup \mathbf{Z}_2 \cup \mathbf{Z}_3$. Os padrões de treinamento são utilizados para obtenção do *bias* e dos valores dos multiplicadores de Lagrange α_i . O conjunto de validação é utilizado como parâmetro para que o algoritmo genético calcule a aptidão de cada indivíduo e, por fim, identifique o melhor indivíduo da população. Já o conjunto de teste (ainda não apresentado ao classificador) avalia a acurácia obtida pelo melhor indivíduo.

Em uma segunda configuração, o conjunto de dados é então, dividido em apenas 2, treinamento e teste, com porcentagens respectivas, 80% e 20%. Nessa configuração, padrões já utilizados pelo classificador para treinamento são utilizados para realizar a validação, o que difere da abordagem anterior. O conjunto de teste funciona de maneira similar, avaliando a acurácia do classificador. Vale salientar que independente da configuração escolhida, o procedimento deverá ser repetido por 20 vezes³ e então a acurácia definida será a média da taxa de acerto de todas as realizações.

É interessante mencionar que o primeiro passo para um algoritmo genético é a definição do conjunto de indivíduos presentes na população inicial⁴, esse procedimento é normalmente realizado de forma aleatória. Cada indivíduo é considerado uma solução para o problema e é constituído por uma cadeia de valores binários, onde cada gene representa a presença (1) ou ausência (0) de determinado padrão no conjunto de vetores-suporte, que é equivalente ao multiplicador de Lagrange associado possuir o valor 0. Já para os casos em que se poda apenas atributos a mesma ideia é válida.

Cada população de uma determinada geração t possui 120 indivíduos, em que cada indivíduo possui g genes cujo tamanho depende do problema em questão. Por exemplo para os métodos de seleção de atributos (Seções 5.1 e 5.2), g terá tamanho igual a quantidade de atributos do conjunto de dados $\mathbf{Z}$. Já para a proposta de padrões e atributos (Seção 5.3),

³ Embaralhando os dados a cada realização.

⁴ Para a proposta FSFGAS-LSSVM o tamanho da população deverá levar em consideração o custo de poda θ

g terá o valor de $N_f + N_p$, onde N_p é a quantidade de padrões do treinamento (i.e., vetores-suporte) e N_f é o número de atributos. A Tabela 10 a seguir ilustra as principais características da abordagem utilizada.

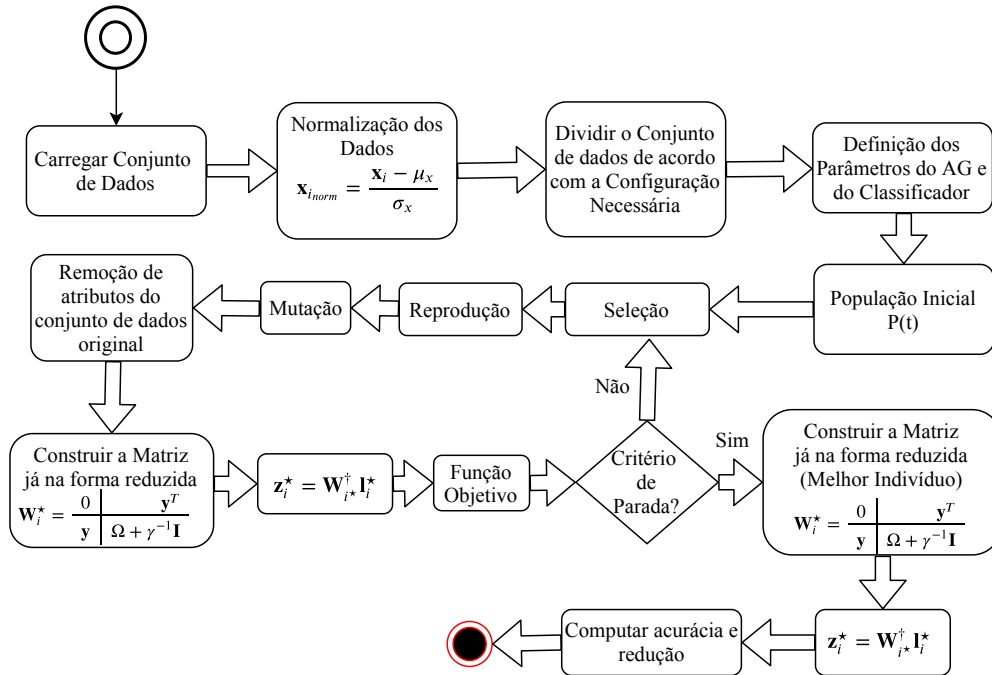
Tabela 10 – Metodologia Utilizada.

Configuração 1	Divisão em 60% para treinamento, 20% para validação e 20% para testes
Configuração 2	Divisão em 80% para treinamento e 20% para testes
Normalização dos Dados	(6.1)
Realizações	20
Gerações - AG	30
População - AG	120

Fonte: Próprio Autor.

Para as propostas desta dissertação o valor de regularização γ foi baseado em trabalhos já realizados como o de Silva e Rocha Neto (2015), Rocha Neto e Barreto (2013), Silva e Rocha Neto (2013), o *kernel* utilizado foi do tipo linear, não necessitando de parâmetro. A escolha de hiperparâmetros para os classificadores é de suma importância para um melhor desempenho, diante disto, basicamente a ideia é utilizar algum método como por exemplo: validação cruzada combinada com busca em grade para encontrar o melhor valor de regularização e utilizar o resultado como parâmetro para os classificadores propostos.

Figura 21 – Metodologia Utilizada para os métodos propostos



Fonte: Próprio Autor.

A Figura 21 representa a sequência de passos utilizada pela abordagem referente a padrões e atributos (5.3), onde primeiramente os atributos (colunas) são removidos do conjunto de dados e posteriormente, os vetores-suporte serão eliminados. Por fim, a taxa de acerto e de redução são calculadas. Uma outra abordagem utilizada (5.1 e 5.2) segue a mesma ideia presente na Figura 21, mas remove apenas os atributos da base de dados, não necessitando de alteração na matriz $\mathbf{W}$, utilizando assim apenas o método da inversa (2.13) para obter os multiplicadores de Lagrange e o *bias*.

Os parâmetros utilizados para fins de comparação são:

- Taxa média de acerto realizado no conjunto de teste.
- Redução de atributos e vetores-suporte.
- Tempo de classificação.

6.2 Resultados

Nesta seção serão apresentados resultados obtidos durante os experimentos, tais como discussões a respeito dos classificadores propostos FPGAS-LSSVM (Proposta 1), FSFGAS-LSSVM (Proposta 2) e PFGAS-LSSVM (Proposta 3).

6.2.1 Resultados para o classificador FPGAS-LSSVM (Proposta 1)

Foram realizadas simulações levando em consideração a configuração 1 (Tabela 10), com a utilização dos conjuntos de dados do mundo real disponíveis no *UCI Repository*¹ como mostra a Tabela 9. Os classificadores LSSVM e PCA-LSSVM foram utilizados para fins de comparação. Pode-se observar na Tabela 11, a acurácia junto com desvio padrão, taxa de poda média (#PR) e o tempo médio de classificação em segundos (#CT) referente a cada classificador e respectivo conjunto de dados, executados no teste após 20 realizações independentes.

Desde que foram realizadas simulações para a proposta (FPGAS-LSSVM), é possível identificar a quantidade de atributos removidos pelo melhor indivíduo da população. Então utiliza-se como objetivo a mesma quantidade de componentes principais para o PCA-LSSVM, ou seja, supondo que o melhor indivíduo para uma determinada execução foi $\mathbf{i} = [1001010]$, então serão mantidos 3 atributos do conjunto de dados original, dessa forma, são selecionados os 3 melhores componentes principais para realizar uma comparação justa obtendo assim a mesma taxa de poda para essas propostas. Além disto, são apresentados os resultados para o PCA-LSSVM ao utilizar diferentes percentuais de informação (isto é, número de componentes principais) na Figura 22, que possui a acurácia versus o número

¹ <http://archive.ics.uci.edu/ml/>.

de componentes principais e porcentagem de informação versus o número de componentes principais.

Cada execução do algoritmo genético, começa gerando uma população de possíveis soluções de forma aleatória. Para as próximas gerações, são mantidos 10% dos melhores indivíduos selecionados pelo elitismo, 80% dos novos indivíduos para reprodução e finalmente, 10% dos novos indivíduos são obtidos por mutação. O algoritmo para quando há convergência, ou seja, a diferença média entre as gerações é de 10^{-5} (valor utilizado como critério de parada do AG) ou quando atingir o número de gerações especificadas (Tabela 10).

Depois de atingir o critério de parada, o melhor indivíduo é selecionado, ou seja, o que obteve melhor acurácia no conjunto de validação $\mathbf{Z}_2$. Analisando os resultados da Tabela 11 é possível observar que o desempenho do classificador proposto com o conjunto de dados reduzido foi semelhante ou até melhor do que o convencional LSSVM e o PCA-LSSVM em alguns casos.

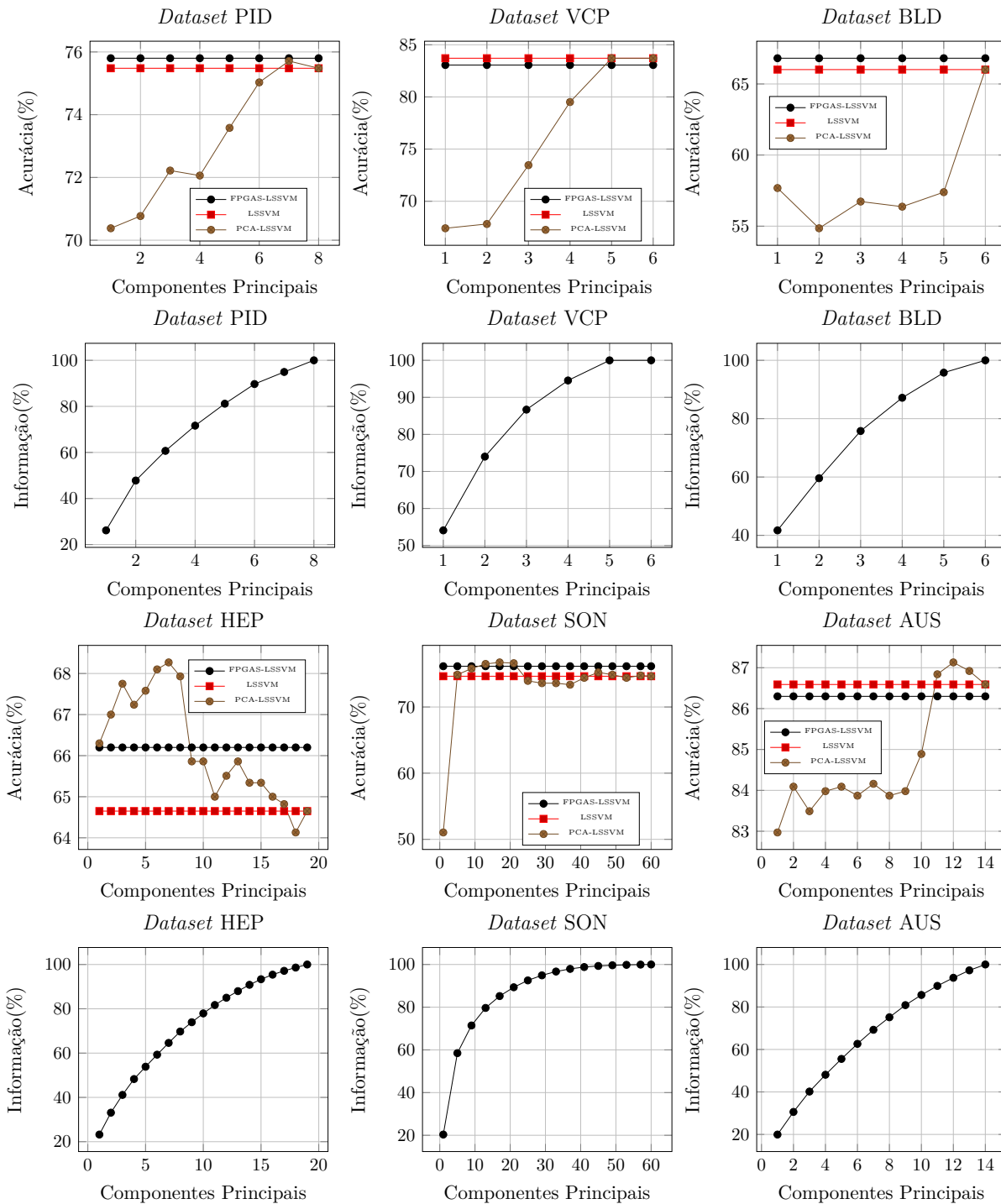
Tabela 11 – Resultados para os classificadores LSSVM, PCA-LSSVM e FPGAS-LSSVM com 60%, 20% e 20% do conjunto de dados para treinamento, validação e teste, respectivamente.

<i>Dataset</i>	<i>Modelo</i>	γ	<i>Acurácia(%)</i>	<i># PR(%)</i>	<i>#CT(s)</i>
PID	LSSVM	0,05	75,48±3,3	-	0,00157
	PCA-LSSVM		73,93±3,2	32,5	0,00120
	FPGAS-LSSVM		75,80±3,2	32,5	0,00134
BLD	LSSVM	0,05	66,01±5,9	-	0,00063
	PCA-LSSVM		59,05±5,9	30,0	0,00023
	FPGAS-LSSVM		66,81±7,0	30,0	0,00054
SON	LSSVM	0,05	74,65±7,6	-	0,00009
	PCA-LSSVM		74,76±6,4	51,7	0,00008
	FPGAS-LSSVM		76,16±5,7	51,7	0,00007
HEP	LSSVM	0,05	64,65±8,7	-	0,00008
	PCA-LSSVM		66,80±7,1	52,8	0,00006
	FPGAS-LSSVM		66,20±6,4	52,8	0,00005
AUS	LSSVM	0,05	86,59±2,3	-	0,00117
	PCA-LSSVM		84,52±2,3	37,1	0,00091
	FPGAS-LSSVM		86,30±2,4	37,1	0,00094
VCP	LSSVM	0,05	83,71±5,7	-	0,00046
	PCA-LSSVM		81,12±6,5	25,0	0,00039
	FPGAS-LSSVM		83,06±5,9	25,0	0,00038

Fonte: Próprio Autor.

Observando a Tabela 11 é possível perceber que a proposta mencionada obtém melhor acurácia para as bases PID, BLD e SON. Também são similares para os demais conjuntos de dados, sendo a acurácia próxima do LSSVM convencional, e na maioria dos casos, melhor que o PCA-LSSVM. Em termos de tempo de classificação, os resultados são semelhantes aos outros.

Figura 22 – Acurácia, número de componentes principais e percentual de informação presentes nos conjuntos de dados PID,VCP,BLD,HEP,SON e AUS.



Fonte: Próprio Autor.

6.2.2 Resultados para o classificador FSFGAS-LSSVM (Proposta 2)

Para essa proposta, foram realizadas simulações levando em conta um conjunto de dados completo $\mathbf{Z}$. Então, são removidos os atributos (colunas) de $\mathbf{Z}$ para obter um novo conjunto de dados denominado $\mathbf{P}$ que foi randomicamente dividido em $\mathbf{P}_1$, $\mathbf{P}_2$ e $\mathbf{P}_3$ com

60%, 20% e 20% dos exemplos para treinamento, validação e teste, respectivamente. Dessa forma, o conjunto de dados $\mathbf{P} = \mathbf{P}_1 \cup \mathbf{P}_2 \cup \mathbf{P}_3$ representa a configuração 1 (Tabela 10).

O classificador LSSVM padrão foi utilizado para fins de comparação. É possível observar na Tabela 12 a acurácia obtida, o desvio padrão, taxa de poda ($\#PR$) e tempo médio de classificação ($\#CT$) em segundos referentes a cada classificador e respectivo *dataset*. Experimentos foram executados em 20 realizações independentes. Observando a Tabela 12 também é possível visualizar que o método proposto obteve melhor acurácia para os conjuntos de dados PID, AUS, BLD, HEP, SON e similar para VCP, visto que a acurácia é bastante próxima com o classificador convencional.

Em termos de tempo de classificação, os resultados são bem similares para os classificadores. Cada execução do AG inicia gerando uma população de possíveis soluções de forma aleatória⁵ semelhante com a abordagem anterior (Seção 6.2.1). Analisando a Tabela 12 é possível visualizar que o método proposto alcançou melhores valores de acurácia(%) para as bases PID, SON, AUS, BLD e HEP sendo similar para VCP.

Após a realização das simulações para a proposta (FSFGAS-LSSVM) são encontrados os resultados médios de poda de atributos. Esses resultados são utilizados como parâmetros para o classificador PCA-LSSVM, motivo pelo qual eles possuem a mesma redução. Portanto, o número de 1's da melhor solução de cada realização será o número de componentes principais nas simulações.

Esta abordagem foi realizada com o intuito de obter uma comparação justa entre os modelos. Visto que nem sempre é possível garantir que a quantidade de atributos removidos seja exatamente igual a taxa de poda, é possível então obter uma taxa de poda aproximada ao valor θ escolhido como mostra a Figura 23 contendo a acurácia(%) e taxa de poda (θ) real para os classificadores e conjuntos de dados.

6.2.3 Resultados para o classificador PFGAS-LSSVM (Proposta 3)

Para esse modelo, as simulações foram realizadas levando em conta um conjunto de dados completo $\mathbf{Z}$. São removidos os atributos (colunas) de $\mathbf{Z}$ para obter um novo conjunto de dados denominado $\mathbf{P}$ que foi randomicamente dividido em $\mathbf{P}_1$, $\mathbf{P}_2$ e $\mathbf{P}_3$, com 60%, 20%, 20% de padrões para treinamento, validação e teste, respectivamente na primeira configuração abordada e 80%, 20% para treinamento e testes na segunda configuração. O processo de poda de atributos é realizado de acordo com a configuração presente no cromossomo $\mathbf{c}_1$ para obter $\mathbf{P}$, então é finalizado o primeiro passo que é a seleção de atributos utilizando algoritmos genéticos.

Sequencialmente, com um novo conjunto de dados reduzido, o processo de treinamento da LSSVM será inicializado com o intuito de remover os padrões (vetores-suporte)

⁵ Levando em consideração o custo de poda θ

Tabela 12 – Resultados para os classificadores FSFGAS-LSSVM, PCA-LSSVM e LSSVM com 60%, 20% e 20% do conjunto de dados completo para treinamento, validação e testes, respectivamente. Acc , θ e $\#PR$ são expressos em porcentagem, $\#CT$ em segundos.

<i>Dataset</i>	<i>Métrica</i>	<i>LSSVM</i>	<i>PCA – LSSVM</i>	<i>FSFGAS – LSSVM</i>		
VCP	Acc	82,90 ±5,7	80,16±6,1	82,50±5,9	81,85±6,2	80,88±6,1
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	33,3	16,6	33,3	50,0
	#CT	0,00028	0,00022	0,00019	0,00018	0,00019
PID	Acc	78,77±3,3	78,38±3,3	78,87 ±3,0	78,38±3,0	77,74±3,3
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	25,0	12,5	25,0	50,0
	#CT	0,00053	0,00054	0,00051	0,00052	0,00051
SON	Acc	73,95±6,7	73,48±6,7	74,76 ±7,1	73,48±7,2	73,02±7,0
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	20,0	10,0	20,0	50,0
	#CT	0,00013	0,00014	0,00012	0,00012	0,00010
AUS	Acc	86,44±2,5	86,37±2,5	86,59 ±2,2	86,41±1,8	86,05±1,7
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	21,4	14,2	21,4	50,0
	#CT	0,00063	0,00064	0,00060	0,00061	0,00060
BLD	Acc	64,92±3,5	56,81±3,3	65,29 ±4,0	63,04±5,0	62,89±5,0
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	33,3	16,6	33,3	50,0
	#CT	0,00021	0,00022	0,00020	0,00020	0,00020
HEP	Acc	67,41±8,3	67,75±8,1	68,10±7,6	69,65 ±7,6	67,93±10,0
	γ	0,05	0,05	0,05	0,05	0,05
	θ	-	20,0	10,0	20,0	50,0
	#PR	-	21,0	10,5	21,0	52,6
	#CT	0,00014	0,00009	0,00013	0,00007	0,00007

Fonte: Próprio Autor.

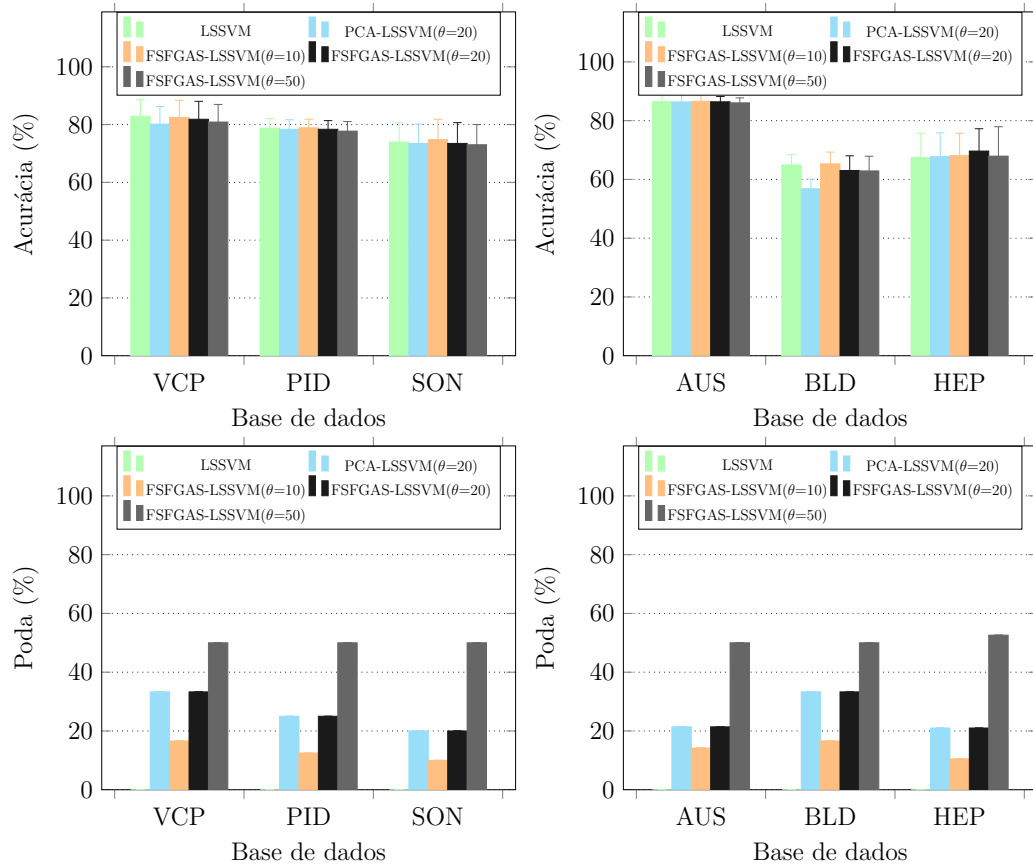
menos relevantes para o problema, alcançando assim a proposta em dois passos para impor esparsidade ao classificador padrão. Geralmente os conjuntos de dados possuem mais padrões que atributos, a remoção de determinada característica causa mais impacto que a remoção de um padrão em particular. O cromossomo $\mathbf{c}_2$ é responsável por determinar quais padrões de treinamento serão mantidos/removidos do conjunto de dados⁶.

É interessante mencionar que a nova matriz $\mathbf{W}^*$ é construída já com tamanho reduzido⁷, diferente de trabalhos anteriores em que primeiro a matriz original $\mathbf{W}$ é obtida para depois realizar a remoção das colunas. A matriz obtida $\mathbf{W}^*$ é não-quadrada, sendo necessário utilizar o método da pseudo-inversa presente na Eq. (2.15) para encontrar a solução $\mathbf{z}$.

⁶ Equivalente a setar os multiplicadores de Lagrange correspondentes para zero.

⁷ Que possui ganhos em armazenamento de memória.

Figura 23 – Acurácia e Poda para todos os conjuntos de dados.



Fonte: Próprio Autor.

Os classificadores P-LSSVM, IP-LSSVM, LSSVM e PCA-LSSVM foram utilizados para fins de comparação. É possível observar na Tabela 13 a acurácia, desvio padrão, redução total (*score*) e média do tempo de classificação em segundos ($\#CT$) referentes a cada classificador e respectivo conjunto de dados. Sendo executados no conjunto de teste após 20 realizações independentes. De acordo com a Tabela 13, a abordagem respeita a configuração com 60% para treinamento, 20% para validação e 20% para testes, de modo que o mesmo conjunto de validação foi utilizado pelo método proposto e o classificador P-LSSVM. Na Tabela 14, uma configuração diferente foi abordada, com 80% dos dados para treinamento e 20% para testes, desde que o classificador IP-LSSVM não necessita de validação.

Após a análise das Tabelas 13 e 14 é interessante salientar que o método proposto conseguiu obter valores de acurácia superiores na maioria dos casos em comparação com os métodos presentes na literatura, bem como conseguiu impor esparsidade considerável baseada na Eq. (5.11), pois em todas as bases de dados os resultados de redução foram os melhores. Os gráficos presentes na Figura 24 representam a relação entre taxa de redução e acurácia com o intuito de comparar o método IP-LSSVM com diferentes taxas em relação a outros métodos já demonstrados. O apêndice A demonstra os gráficos de convergência.

Tabela 13 – Resultados para os classificadores LSSVM, PFGAS-LSSVM, P-LSSVM e PCA-LSSVM com 60%, 20% e 20% do conjunto de treinamento, validação e testes, respectivamente.

<i>Dataset</i>	Modelo	γ	Acurácia(%)	<i>score</i> (%)	#CT(s)
PID	LSSVM	0,05	75,51±2,67	-	0,00057
	P-LSSVM	0,05	75,51±2,52	31,13	0,00019
	PCA-LSSVM	0,05	73,41±3,54	33,75	0,00054
	PFGAS-LSSVM	0,05	75,74±3,47	66,26	0,00027
VCP	LSSVM	0,05	83,79±5,37	-	0,00020
	P-LSSVM	0,05	82,17±6,44	30,10	0,00006
	PCA-LSSVM	0,05	82,25±6,14	23,33	0,00021
	PFGAS-LSSVM	0,05	84,91±3,89	61,42	0,00005
BLD	LSSVM	0,05	66,08±4,66	-	0,00024
	P-LSSVM	0,05	62,97±6,62	11,57	0,00010
	PCA-LSSVM	0,05	57,10±6,15	26,66	0,00025
	PFGAS-LSSVM	0,05	67,17±4,49	63,98	0,00006
HEP	LSSVM	0,05	67,06±8,02	-	0,00008
	P-LSSVM	0,05	66,89±9,32	21,88	0,00003
	PCA-LSSVM	0,05	70,00±9,01	51,05	0,00007
	PFGAS-LSSVM	0,05	67,41±9,25	74,90	0,00003
SON	LSSVM	0,05	75,23±5,95	-	0,00014
	P-LSSVM	0,05	72,67±6,57	11,00	0,00005
	PCA-LSSVM	0,05	75,23±5,55	49,91	0,00013
	PFGAS-LSSVM	0,05	76,04±7,47	75,14	0,00004
AUS	LSSVM	0,05	86,19±2,79	-	0,00050
	P-LSSVM	0,05	85,50±3,03	58,59	0,00012
	PCA-LSSVM	0,05	85,21±3,21	33,21	0,00049
	PFGAS-LSSVM	0,05	86,37±2,78	66,50	0,00023

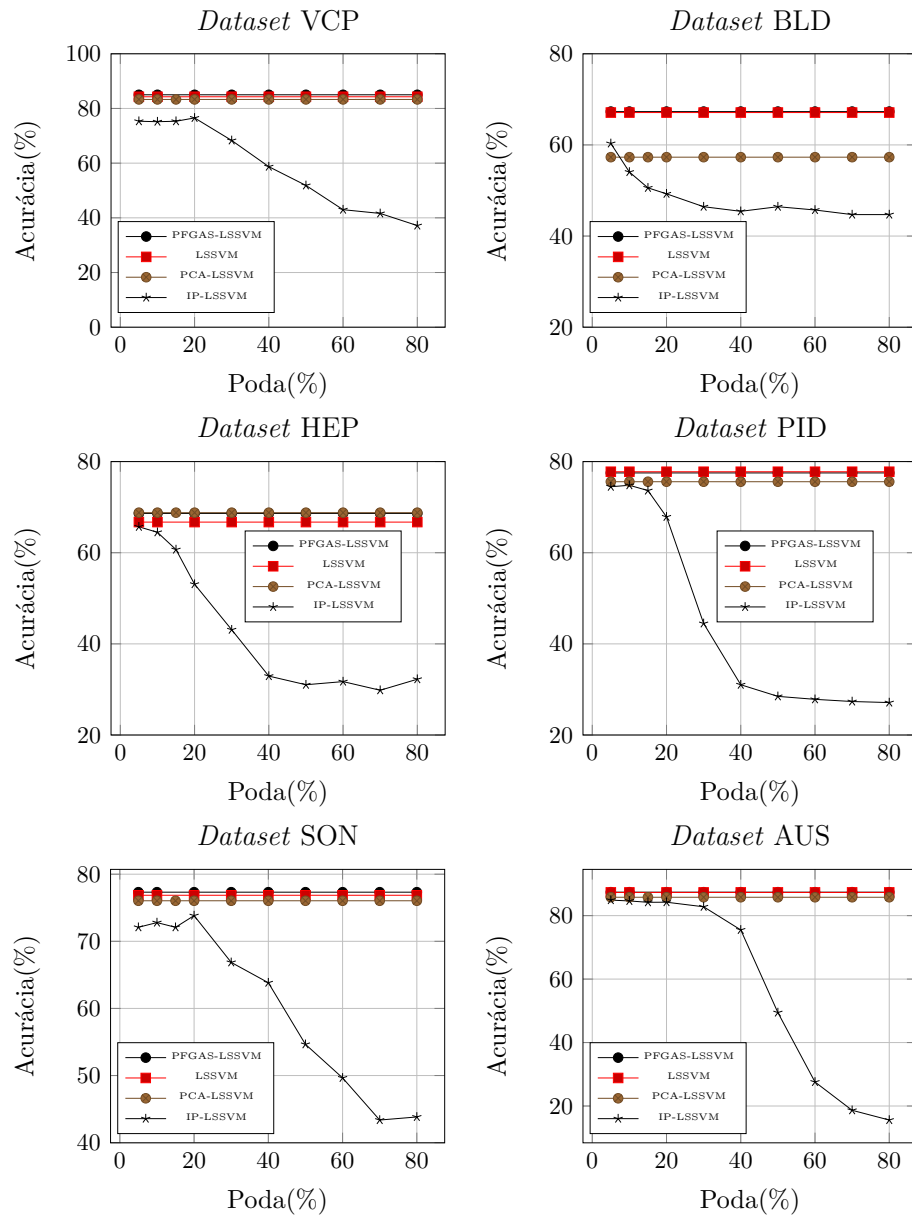
Fonte: Próprio Autor.

Tabela 14 – Resultados para os classificadores LSSVM, PFGAS-LSSVM, IP-LSSVM e PCA-LSSVM com 80% e 20% do conjunto de dados para treinamento e teste, respectivamente.

<i>Dataset</i>	Modelo	γ	Acurácia(%)	<i>score</i> (%)	#CT(s)
PID	LSSVM	0,05	77,77 $\pm$ 3,34	-	0,00067
	IP-LSSVM	0,05	74,80 $\pm$ 3,13	10,0	0,00042
			67,87 $\pm$ 12,3	20,0	0,00041
			44,48 $\pm$ 13,7	30,0	0,00040
	PCA-LSSVM	0,05	75,58 $\pm$ 3,09	31,8	0,00065
	PFGAS-LSSVM	0,05	77,51 $\pm$ 3,26	66,1	0,00027
VCP	LSSVM	0,05	84,27 $\pm$ 4,87	-	0,00027
	IP-LSSVM	0,05	75,16 $\pm$ 5,19	10,0	0,00009
			76,45 $\pm$ 5,11	20,0	0,00009
			68,30 $\pm$ 17,4	30,0	0,00009
	PCA-LSSVM	0,05	83,30 $\pm$ 5,36	20,8	0,00027
	PFGAS-LSSVM	0,05	85,00 $\pm$ 4,41	61,0	0,00008
BLD	LSSVM	0,05	67,10 $\pm$ 6,48	-	0,00030
	IP-LSSVM	0,05	54,05 $\pm$ 5,79	10,0	0,00012
			49,27 $\pm$ 4,93	20,0	0,00013
			46,44 $\pm$ 5,03	30,0	0,00013
	PCA-LSSVM	0,05	57,31 $\pm$ 5,76	26,6	0,00030
	PFGAS-LSSVM	0,05	67,31 $\pm$ 6,31	63,0	0,00011
HEP	LSSVM	0,05	66,72 $\pm$ 8,75	-	0,00010
	IP-LSSVM	0,05	64,48 $\pm$ 7,75	10,0	0,00005
			53,10 $\pm$ 16,2	20,0	0,00005
			43,10 $\pm$ 17,2	30,0	0,00003
	PCA-LSSVM	0,05	68,79 $\pm$ 7,62	48,1	0,00006
	PFGAS-LSSVM	0,05	68,62 $\pm$ 7,57	74,0	0,00003
SON	LSSVM	0,05	76,86 $\pm$ 5,36	-	0,00017
	IP-LSSVM	0,05	72,79 $\pm$ 6,58	10,0	0,00006
			73,83 $\pm$ 6,48	20,0	0,00006
			66,86 $\pm$ 9,54	30,0	0,00006
	PCA-LSSVM	0,05	76,04 $\pm$ 6,13	45,8	0,00015
	PFGAS-LSSVM	0,05	77,32 $\pm$ 7,11	71,5	0,00005
AUS	LSSVM	0,05	87,31 $\pm$ 3,60	-	0,00058
	IP-LSSVM	0,05	84,63 $\pm$ 2,75	10,0	0,00035
			84,23 $\pm$ 3,34	20,0	0,00035
			82,79 $\pm$ 3,13	30,0	0,00034
	PCA-LSSVM	0,05	85,79 $\pm$ 3,83	33,5	0,00058
	PFGAS-LSSVM	0,05	87,39 $\pm$ 3,53	67,2	0,00028

Fonte: Próprio Autor.

Figura 24 – Acurácia e percentual de poda presentes nos conjuntos de dados.



Fonte: Próprio Autor.

6.3 Conclusão

Neste capítulo foi apresentada a metodologia presente durante as simulações, bem como as duas abordagens de divisão de dados necessárias para a realização de comparações entre os modelos. As descrições das bases de dados utilizadas: *Vertebral Column Pathologies*, *Pima Indians Diabets*, *Bupa Liver Disorders*, *Australian Credit Approval*, *Sonar* e *Hepatitis* são demonstradas em união com as propriedades como quantidade de atributos e padrões.

Apresenta-se ainda todas as métricas de desempenho utilizadas para fins de comparação das propostas expostas com os métodos já presentes na literatura, além dos resultados em gráficos e tabelas para abordagens em que se poda atributos e/ou padrões. Em alguns casos foi necessário obter um percentual de divisão diferente do conjunto de treinamento e teste, para melhor adequação aos métodos P-LSSVM e IP-LSSVM.

7 Conclusões e Trabalhos Futuros

Como mencionado anteriormente, o classificador LSSVM possui vantagem em relação ao SVM no quesito de encontrar a solução por meio de equações lineares ao invés de utilizar programação quadrática. Entretanto, uma característica importante foi perdida: a detecção automática de vetores suporte e consequentemente o armazenamento de todo o conjunto de treinamento causando falta de esparsidade. Fator que se houvesse presente na LSSVM seria bastante proveitoso em diversos setores, onde seria plausível a representação de uma base de dados demasiadamente grande por meio de uma baixa quantidade de vetores suporte.

Vale salientar que um conjunto reduzido de vetores suporte pode representar as características inerentes do conjunto de dados. Ao longo dos anos, vários trabalhos foram propostos pela literatura com o intuito de solucionar o problema acima mencionado. Algoritmos Genéticos são bastante utilizados para encontrar os parâmetros intrínsecos do classificador e/ou *kernel*, porém nesta dissertação essa meta-heurística é utilizada para encontrar a melhor solução que imponha esparsidade ao classificador mantendo ou até melhorando a acurácia em comparação com o classificador convencional.

Portanto, foram propostos três novos métodos de imposição de esparsidade à LSSVM. A primeira proposta possui objetivo simples de poda de atributos para o classificador LSSVM denominada *Feature Pruning Genetic Algorithm for Sparse LSSVM* (FPGAS-LSSVM), que realiza seleção de atributos por meio de algoritmos genéticos, a fim de identificar e remover os atributos menos relevantes para o problema mantendo praticamente a mesma acurácia e eventualmente, melhorando-a. É interessante ressaltar que a remoção de atributos geralmente possui maior impacto que a remoção de padrões.

A segunda proposta intitulada *Fixed Size Feature Genetic Algorithm for Sparse LSSVM* (FSFGAS-LSSVM), tem como objetivo principal a eliminação de atributos, semelhante à primeira proposta, mas com a adição de um parâmetro θ para determinar o percentual de atributos que serão removidos. Diante disto, os operadores genéticos padrões sofreram alterações para garantir a mesma quantidade de '0's e '1's ao longo das gerações, o que categoriza quais atributos vão permanecer e quais serão eliminados. De todo modo ainda existe o objetivo de encontrar a melhor acurácia possível para cada parâmetro selecionado. Vale ressaltar que na reprodução uma abordagem de tradução de codificação entre binária-inteira e vice-versa foi utilizada para garantir que a quantidade fixa de atributos removidos fosse respeitada.

A terceira proposta denominada *Patterns and Features Genetic Algorithm for Sparse LSSVM* (PFGAS-LSSVM) combina as vantagens da seleção de atributos em

conjunto com a poda de padrões de treinamento, obtendo assim reduções consideráveis em comparação com outros classificadores presentes na literatura. Com base nisto, uma função de *score* foi desenvolvida com o propósito de concentrar as reduções de atributos/padrões em um percentual para que possa ser realizada a comparação com outros métodos. Uma característica bastante importante dessa abordagem é a construção da matriz $\mathbf{W}$ já em tamanho reduzido ($\mathbf{W}^*$), diferente de trabalhos anteriores onde a matriz é obtida completa pra posteriormente remover as colunas. Por fim, foi possível notar, após as simulações, os consideráveis resultados obtidos tanto em questões de acurácia quanto em poda.

7.1 Contribuições deste Trabalho

As contribuições apresentadas neste trabalho estão relacionadas aos temas: Máquinas de Vetores Suporte por Mínimos Quadrados, Algoritmos Genéticos, Métodos de poda relacionados a Padrões e Atributos.

O Capítulo 2 aborda as fundamentações teóricas da LSSVM, para o melhor entendimento do leitor a respeito do classificador abordado. No Capítulo 5 os classificadores FPGAS-LSSVM, FSFGAS-LSSVM e PFGAS-LSSVM são apresentados, bem como a metodologia utilizada com o contexto de algoritmos genéticos para obtenções de soluções esparsas por meio de seleção de atributos e/ou padrões e com valores de acurácia competitivos com os classificadores convencionais. Além disso, uma nova fórmula denominada *score* foi desenvolvida a fim de obter os valores de poda em percentuais, unindo a abordagem de construção da matriz $\mathbf{W}^*$ já em tamanho reduzido.

Por fim, o Capítulo 6 demonstra os resultados de todas as simulações, bem como valores de acurácia, taxas de poda e gráficos. Assegurando que as propostas apresentadas cumprem os objetivos inicialmente definidos.

7.2 Trabalhos Futuros

Como trabalhos futuros pode-se recomendar a utilização de *kernels* não lineares como por exemplo *Radial Basis Function* (RBF) no classificador LSSVM, bem como a geração de uma população inicial de forma otimizada, ou seja, mais próxima da melhor solução.

Outro trabalho sugerido é a realização no âmbito de seleção de atributos da abordagem multi-objetiva para obtenção de soluções esparsas. Já em relação ao método que poda padrões e atributos seria interessante a versão de tamanho fixo.

Sugere-se ainda a modificação dos métodos propostos para problemas multiclases, o que amplia a gama de conjuntos de dados que podem ser utilizados, pois atualmente utiliza-se apenas problemas binários.

Referências

ALENCAR, A. S.; ROCHA NETO, A. R.; GOMES, J. P. P. A new pruning method for extreme learning machines via genetic algorithms. *Applied Soft Computing*, Elsevier, v. 44, p. 101–107, 2016. Citado na página 30.

ÁLVAREZ, D. et al. Feature selection from nocturnal oximetry using genetic algorithms to assist in obstructive sleep apnoea diagnosis. *Medical engineering & physics*, Elsevier, v. 34, n. 8, p. 1049–1057, 2012. Citado na página 30.

ALVES, S. S.; OLIVEIRA, S. A.; ROCHA NETO, A. R. A recursive genetic algorithm-based approach for educational timetabling problems. In: *Designing with Computational Intelligence*. Switzerland: Springer, 2017. v. 664, p. 161–175. Citado na página 53.

ARAÚJO, P. H. M.; ROCHA NETO, A. R. Sparse least squares support vector machines based on genetic algorithms: A feature selection approach. In: SPRINGER. *International Work-Conference on Artificial Neural Networks*. Gran Canaria, Spain, 2019. v. 11507, p. 500–511. Citado 3 vezes nas páginas 31, 69 e 78.

BARBER, D. The use of principal components in the quantitative analysis of gamma camera dynamic studies. *Physics in Medicine & Biology*, IOP Publishing, v. 25, n. 2, p. 283, 1980. Citado na página 63.

BARBOSA, G. S.; COSTA, L. da S.; ROCHA NETO, A. R. A new genetic algorithm-based pruning approach for optimum-path forest. In: IEEE. *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*. São Paulo, SP, 2018. p. 13–18. Citado na página 30.

BELHUMEUR, P. N.; HESPANHA, J. P.; KRIEGMAN, D. J. *Eigenfaces vs. fisherfaces: Recognition using class specific linear projection*. New Haven, 1997. Citado na página 68.

BERTSEKAS, D. P.; RHEINBOLDT, W. Constrained optimization and lagrange multiplier methods. Academic Press, 1982. Citado na página 38.

BISHOP, C. M. *Pattern recognition and machine learning*. New York: springer, 2006. Citado na página 29.

BRINDLE, A. Genetic algorithms for function optimization. 1980. Citado na página 51.

ÇALIŞIR, D.; DOGANTEKIN, E. A new intelligent hepatitis diagnosis system: Pca-lssvm. *Expert Systems with Applications*, Elsevier, v. 38, n. 8, p. 10705–10708, 2011. Citado 2 vezes nas páginas 67 e 78.

CAO, B. et al. Feature selection in a kernel space. In: ACM. *Proceedings of the 24th international conference on Machine learning*. Corvallis, OR, 2007. p. 121–128. Citado na página 63.

CARVALHO, B. *Novas estratégias para detecção automática de vetores de suporte em least squares support vector machines*. Tese (Doutorado) — Dissertação (Mestrado em Engenharia Elétrica)-Universidade Federal de Minas . . . , 2005. Citado na página 38.

- CARVALHO, B.; BRAGA, A. Ip-lssvm: A two-step sparse classifier. *Pattern Recognition Letters*, Elsevier, v. 30, n. 16, p. 1507–1515, 2009. Citado 6 vezes nas páginas 30, 36, 37, 57, 59 e 78.
- DU, Q.; FOWLER, J. E. Hyperspectral image compression using jpeg2000 and principal component analysis. *IEEE Geoscience and Remote sensing letters*, IEEE, v. 4, n. 2, p. 201–205, 2007. Citado na página 63.
- DUDOIT, S.; FRIDLYAND, J.; SPEED, T. P. Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American statistical association*, Taylor & Francis, v. 97, n. 457, p. 77–87, 2002. Citado na página 68.
- EIBEN, A. E.; SMITH, J. E. *Introduction to evolutionary computing*. 2. ed. Amsterdam: Springer, 2003. v. 53. Citado na página 73.
- FARAGÓ, P. et al. An evolutionary multi-objective optimization approach to design the sound processor of a hearing aid. In: SPRINGER. *International Conference on Advancements of Medicine and Health Care through Technology; 5th–7th June 2014*. Cluj-Napoca, Romania, 2014. v. 44, p. 181–186. Citado na página 30.
- FAYYAD, U.; PIATETSKY-SHAPIO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37–37, 1996. Citado na página 62.
- FLETCHER, R. *Practical methods of optimization*. 2. ed. New York: John Wiley & Sons, 2013. Citado na página 39.
- FOGEL, L. J.; OWENS, A. J.; WALSH, M. J. *Artificial intelligence through simulated evolution*. New York, NY: Wiley, 1966. Citado na página 29.
- FUKUAGA, K. Introduction to statistical pattern classification. *Patt. Recognit*, v. 30, n. 7, p. 1145–1149, 1990. Citado na página 67.
- GEEBELEN, D.; SUYKENS, J. A.; VANDEWALLE, J. Reducing the number of support vectors of svm classifiers using the smoothed separable case approximation. *IEEE Transactions on Neural Networks and Learning Systems*, IEEE, v. 23, n. 4, p. 682–688, 2012. Citado na página 31.
- GESTEL, T. V. et al. Financial time series prediction using least squares support vector machines within the evidence framework. *IEEE Transactions on neural networks*, IEEE, v. 12, n. 4, p. 809–821, 2001. Citado na página 30.
- GESTEL, T. V. et al. Multiclass ls-svms: Moderated outputs and coding-decoding schemes. *Neural processing letters*, Springer, v. 15, n. 1, p. 45–58, 2002. Citado na página 35.
- GOLDBERG, D. E. *Genetic Algorithms in search, optimization and machine learning*. Boston, MA, USA: Addison-Wesley, 1989. Citado 2 vezes nas páginas 54 e 60.
- GOLDBERG, D. E.; DEB, K. A comparative analysis of selection schemes used in genetic algorithms. In: *Foundations of genetic algorithms*. Urbana, IL: Elsevier, 1991. v. 1, p. 69–93. Citado na página 51.
- GOLDBERG, D. E.; HOLLAND, J. H. Genetic algorithms and machine learning. *Machine learning*, Springer, v. 3, n. 2, p. 95–99, 1988. Citado 2 vezes nas páginas 30 e 53.

- HAMDANI, T. M. et al. Hierarchical genetic algorithm with new evaluation function and bi-coded representation for the selection of features considering their confidence rate. *Applied Soft Computing*, Elsevier, v. 11, n. 2, p. 2501–2509, 2011. Citado na página 48.
- HAN, J.; PEI, J.; KAMBER, M. *Data mining: concepts and techniques*. Waltham, MA: Elsevier, 2011. Citado na página 83.
- HAUPT, R. L.; HAUPT, S. E. *Practical genetic algorithms*. New Jersey: Wiley-interscience, 2004. Citado na página 60.
- HAYKIN, S. S. et al. *Neural networks and learning machines*. New Jersey: New York: Prentice Hall, 2009. Citado 3 vezes nas páginas 63, 64 e 65.
- HOLLAND, J. H. et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Cambridge, Massachusetts: MIT press, 1992. Citado 2 vezes nas páginas 29 e 54.
- IZZO, D.; SPRAGUE, C.; TAILOR, D. Machine learning and evolutionary techniques in interplanetary trajectory design. 02 2018. Citado na página 29.
- JAUHARI, S.; RIZVI, S. Mining gene expression data focusing cancer therapeutics: a digest. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, IEEE Computer Society Press, v. 11, n. 3, p. 533–547, 2014. Citado na página 30.
- JOLLIFFE, I. T. *Principal Component Analysis*. New York: Springer-Verlag, 2002. Citado na página 63.
- JONG, K. A. D. Analysis of the behavior of a class of genetic adaptive systems. 1975. Citado na página 51.
- KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. *Artificial intelligence*, Elsevier, v. 97, n. 1-2, p. 273–324, 1997. Citado 2 vezes nas páginas 62 e 63.
- KOHONEN, T. *Self-Organizing Maps*. New York: Springer-Verlag Berlin Heidelberg, 2001. Citado na página 57.
- LACERDA, E. G. de; CARVALHO, A. de. Introdução aos algoritmos genéticos. *Sistemas inteligentes: aplicações a recursos hídricos e ciências ambientais*, v. 1, p. 99–148, 1999. Citado 2 vezes nas páginas 47 e 54.
- LEE, H. D. *Seleção de atributos importantes para a extração de conhecimento de bases de dados*. Tese (Doutorado) — Universidade de São Paulo, 2005. Citado 3 vezes nas páginas 31, 62 e 63.
- LEE, Y.-J.; MANGASARIAN, O. L. Rsvm: Reduced support vector machines. In: SIAM. *Proceedings of the 2001 SIAM International Conference on Data Mining*. Chicago, IL, 2001. p. 1–17. Citado na página 41.
- LEE, Y.-J.; MANGASARIAN, O. L. Ssvm: A smooth support vector machine for classification. *Computational optimization and Applications*, Springer, v. 20, n. 1, p. 5–22, 2001. Citado na página 42.
- MICHALEWICZ, Z. *Genetic algorithms+ data structures= evolution programs*. New York: Springer Science & Business Media, 1996. Citado 2 vezes nas páginas 46 e 55.

- MITCHELL, M. *An introduction to genetic algorithms*. Cambridge, Massachusetts: MIT press, 1998. Citado 2 vezes nas páginas 30 e 54.
- MOORE, E. H. On the reciprocal of the general algebraic matrix. *Bull. Am. Math. Soc.*, v. 26, p. 394–395, 1920. Citado na página 42.
- MUSTAFA, M. et al. Reactive power tracing in pool-based power system utilising the hybrid genetic algorithm and least squares support vector machine. *IET generation, transmission & distribution*, IET, v. 6, n. 2, p. 133–141, 2012. Citado na página 60.
- OLIVEIRA, D. F. *Análise de Métodos de Redução de Dimensionalidade para Classificação de Padrões*. Monografia — IFCE, 2016. Citado na página 64.
- OLIVEIRA, S. A.; GOMES, J. P.; ROCHA NETO, A. R. Sparse least-squares support vector machines via accelerated segmented test: A dual approach. *Neurocomputing*, Elsevier, v. 321, p. 308–320, 2018. Citado na página 36.
- PALENCIA, A. E. R.; DELGADILLO, G. E. M. A computer application for a bus body assembly line using genetic algorithms. *International Journal of Production Economics*, Elsevier, v. 140, n. 1, p. 431–438, 2012. Citado na página 30.
- PENCHEVA, T.; ATANASSOV, K.; SHANNON, A. Modelling of a roulette wheel selection operator in genetic algorithms using generalized nets. *Int. J. Bioautomation*, v. 13, n. 4, p. 257–264, 2009. Citado na página 49.
- PENROSE, R. A generalized inverse for matrices. In: CAMBRIDGE UNIVERSITY PRESS. *Mathematical proceedings of the Cambridge philosophical society*. [S.l.], 1955. v. 51, n. 3, p. 406–413. Citado na página 42.
- PEREIRA, F. *Estudo das interações entre evolução e aprendizagem em ambientes de computação evolucionária*. Tese (Doutorado) — PhD Thesis, Universidade de Coimbra, 2002. Citado na página 29.
- PERES, R. T.; PEDREIRA, C. E. Generalized risk zone: Selecting observations for classification. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 31, n. 7, p. 1331–1337, 2009. Citado na página 31.
- RAZALI, N. M.; GERAGHTY, J. et al. Genetic algorithm performance with different selection strategies in solving tsp. In: INTERNATIONAL ASSOCIATION OF ENGINEERS HONG KONG. *Proceedings of the world congress on engineering*. London, U.K., 2011. v. 2, n. 1, p. 1–6. Citado 2 vezes nas páginas 49 e 50.
- ROCHA NETO, A. *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 ..., 2011. Citado na página 66.
- ROCHA NETO, A. R.; BARRETO, G. A. Opposite maps: Vector quantization algorithms for building reduced-set svm and lssvm classifiers. *Neural processing letters*, Springer, v. 37, n. 1, p. 3–19, 2013. Citado 2 vezes nas páginas 57 e 85.

- SAMADZADEGAN, F.; SOLEYMANI, A.; ABBASPOUR, R. A. Evaluation of genetic algorithms for tuning svm parameters in multi-class problems. In: IEEE. *2010 11th International Symposium on Computational Intelligence and Informatics (CINTI)*. Budapest, Hungary, 2010. p. 323–328. Citado na página 60.
- SILVA, D. A. *Novas abordagens de poda em máquinas de vetores-suporte por mínimos quadrados via computação evolutiva*. 125 p. Dissertação (Mestrado) — Instituto Federal de Educação, Ciência e Tecnologia do Ceará IFCE, Fortaleza, 2015. Citado na página 39.
- SILVA, D. A.; ROCHA NETO, A. R. Multi-objective genetic algorithms for sparse least square support vector machines. In: SPRINGER. *International Conference on Intelligent Data Engineering and Automated Learning*. Salamanca, Spain, 2014. p. 158–166. Citado na página 61.
- SILVA, D. A.; ROCHA NETO, A. R. A genetic algorithms-based lssvm classifier for fixed-size set of support vectors. In: SPRINGER. *International Work-Conference on Artificial Neural Networks*. Palma de Mallorca, Spain, 2015. p. 127–141. Citado 5 vezes nas páginas 30, 60, 62, 73 e 78.
- SILVA, D. A.; SILVA, J. P.; ROCHA NETO, A. R. Novel approaches using evolutionary computation for sparse least square support vector machines. *Neurocomputing*, Elsevier, v. 168, p. 908–916, 2015. Citado 4 vezes nas páginas 30, 60, 78 e 85.
- SILVA, J. P.; ROCHA NETO, A. R. da. Sparse least squares support vector machines via genetic algorithms. In: IEEE. *2013 BRICS Congress on Computational Intelligence and 11th Brazilian Congress on Computational Intelligence*. Ipojuca, PE, 2013. p. 248–253. Citado 5 vezes nas páginas 30, 60, 61, 78 e 85.
- SILVA, S. F. d. *Seleção de características por meio de algoritmos genéticos para aprimoramento de rankings e de modelos de classificação*. Tese (Doutorado) — Universidade de São Paulo, 2011. Citado na página 63.
- SMITH, L. I. *A tutorial on principal components analysis*. Dunedin, Otago, New Zealand, 2002. Citado na página 64.
- SPEARS, W. M.; JONG, K. D. D. *On the virtues of parameterized uniform crossover*. San Diego, 1991. Citado na página 52.
- SUYKENS, J. A.; LUKAS, L.; VANDEWALLE, J. Sparse approximation using least squares support vector machines. In: IEEE. *2000 IEEE International Symposium on Circuits and Systems. Emerging Technologies for the 21st Century. Proceedings (IEEE Cat No. 00CH36353)*. Geneva, Switzerland, 2000. v. 2, p. 757–760. Citado 2 vezes nas páginas 57 e 58.
- SUYKENS, J. A.; VANDEWALLE, J. Least squares support vector machine classifiers. *Neural processing letters*, Springer, v. 9, n. 3, p. 293–300, 1999. Citado 4 vezes nas páginas 30, 35, 39 e 40.
- SUYKENS, J. A.; VANDEWALLE, J. Multiclass least squares support vector machines. In: *IJCNN*. Washington, DC, USA: [s.n.], 1999. v. 2, p. 900–903. Citado na página 35.
- SUYKENS, J. A. K.; GESTEL, T. V.; BRABANTER, J. D. *Least squares support vector machines*. Singapore: World Scientific, 2002. Citado na página 40.

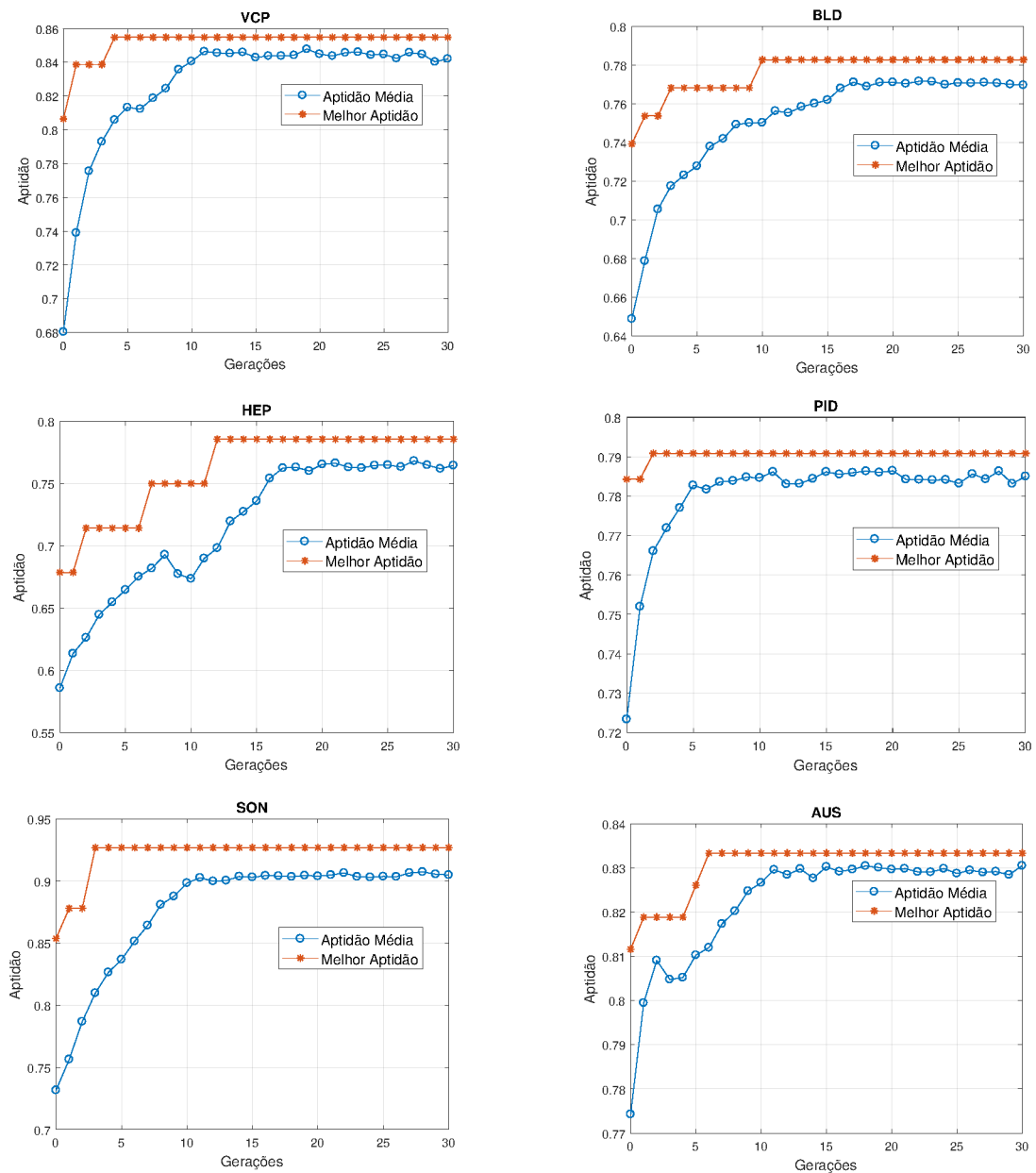
- THEODORIDIS, S.; KOUTROUMBAS, K. *Pattern Recognition*. Fourth edition. Burlington, MA, USA: Academic Press, New York, 2009. v. 4. Citado na página 29.
- THOMAZ, C. E.; GIRALDI, G. A. A new ranking method for principal components analysis and its application to face image analysis. *Image and Vision Computing*, Elsevier, v. 28, n. 6, p. 902–913, 2010. Citado na página 63.
- VALYON, J.; HORVÁTH, G. A sparse least squares support vector machine classifier. In: IEEE. *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*. Bruges, Belgium, 2004. v. 1, p. 543–548. Citado 3 vezes nas páginas 58, 59 e 61.
- VAPNIK, V. N. The nature of statistical learning. *Theory*, springer, 1995. Citado 2 vezes nas páginas 35 e 39.
- WRIGHT, A. H. Genetic algorithms for real parameter optimization. In: *Foundations of genetic algorithms*. Missoula, Montana: Elsevier, 1991. v. 1, p. 205–218. Citado 2 vezes nas páginas 51 e 52.
- YANG, L. et al. Sparse least square support vector machine via coupled compressive pruning. *Neurocomputing*, Elsevier, v. 131, p. 77–86, 2014. Citado na página 30.
- YE, J.; JANARDAN, R.; LI, Q. Two-dimensional linear discriminant analysis. In: *Advances in neural information processing systems*. Vancouver, Canada: [s.n.], 2005. p. 1569–1576. Citado na página 68.
- YUANCHENG, L.; TINGJIAN, F.; ERKENG, Y. Short-term electrical load forecasting using least squares support vector machines. In: IEEE. *Proceedings. International Conference on Power System Technology*. Kunming, China, 2002. v. 1, p. 230–233. Citado na página 30.
- ZHAO, W. et al. Discriminant analysis of principal components for face recognition. In: *Face Recognition: From Theory to Applications*. Scotland, UK: Springer, Berlin, Heidelberg, 1998. p. 73–85. Citado na página 63.

Apêndices

APÊNDICE A – Gráficos de Convergência

Este apêndice tem como objetivo a demonstração da convergência do algoritmo genético ao longo das gerações possuindo como base a aptidão obtida a partir do conjunto de validação na proposta 3 (Seção 5.3). A Figura 25 demonstra o valor médio de aptidão obtido para cada geração, bem como o melhor valor de aptidão (melhor indivíduo) para cada geração. Vale salientar que os gráficos foram obtidos por uma realização.

Figura 25 – Gráficos de convergência para os conjuntos de dados.



Fonte: Próprio autor.